Graph Drawing
Beyond the Beaten Tracks

Dissertation
der Mathematisch-Naturwissenschaftlichen Fakultät
der Eberhard Karls Universität Tübingen
zur Erlangung des Grades eines
Doktors der Naturwissenschaften
(Dr. rer. nat.)

vorgelegt von
Henry Förster (M.Sc.)
aus Wippra

Tübingen
2020

Gedruckt mit Genehmigung der Mathematisch-Naturwissenschaftlichen Fakultät der Eberhard Karls Universität Tübingen.

Tag der mündlichen Qualifikation:	30.09.2020
Stellvertretender Dekan:	Prof. Dr. József Fortágh
1. Berichterstatter:	Prof. Dr. Michael Kaufmann
2. Berichterstatter:	Prof. Dr. Klaus-Jörn Lange
3. Berichterstatter:	Prof. Dr. Markus Chimani

Abstract

Graph drawing is a well-established research area in theoretical computer science with an active research community that studies the embedding of graphs on surfaces such as the Euclidean plane. While traditional results stem from discrete maths, graph embeddings have become more widespread with the emergence of computing technology. Two important applications are the design of computer chips and various diagram types such as UML and BPMN whose development was possible due to the ubiquity of computers in professional environments. While the traditional settings emerging from these newer applications are by now well studied, research has started to focus on advanced settings whose analysis becomes more technically involved. One of the most important research directions of this type is known as graph drawing beyond planarity and studies drawings of graphs where the types of edge intersections are restricted. In this thesis, we consider three such settings that go beyond the beaten tracks.

The most important drawing model in diagramming applications are orthogonal drawings in which edges are represented by polylines whose segments are axis-aligned. Unsurprisingly, there exists a large body of research focusing mostly on planar orthogonal drawings. We investigate two settings that have been proposed in the literature that extend beyond this traditional model: In the smooth orthogonal drawing model, bends are replaced by circular arcs, while in octilinear drawings, segments of polylines may additionally have slopes ± 1. In the first part of this thesis, we show several new results for these "beyond orthogonal" drawing styles: First, we characterize the relationship between the classes of graphs that admit smooth orthogonal and octilinear drawings, respectively. Second, we prove that techniques which diverge from the traditional ones used for orthogonal graph drawings are needed to compute smooth orthogonal and octilinear drawings of minimal curve complexity. Third, we give a drawing algorithm that guarantees several desirable properties. Finally, we also extend the study of orthogonal and smooth orthogonal drawings to the 1-planar setting where edges may be intersected at most once. Here, we provide algorithms which compute drawings whose curve complexity we prove to be worst-case optimal in almost all scenarios.

A graph drawing type that emerged from applications in VLSI chip design are linear layouts. In a linear layout, vertices are totally ordered on a line called spine while edges are drawn entirely above the spine. In a stack layout, the edges are additionally colored so that no two edges of the same color class intersect. This model has been deeply studied especially for planar graphs. The most important result in this field is that four colors suffice for the edges of any planar graph. A related model called queue layouts, however, has proven to be much more difficult to analyze. In a queue layout, edges are also colored, in contrast to stack layouts however, edges of the same color class may intersect but not nest. For more than 25 years the conjecture that a constant number of colors suffices for planar graphs had been open. In the second part of the thesis, we show that for planar

graphs of bounded degree this is indeed true. We point out that the result has been generalized to all planar graphs in the meanwhile. We also consider another drawing style that extends beyond the capabilities of stack layouts called arc diagrams. In contrast to stack layouts, edges in an arc diagram may be drawn above and below the spine and may even intersect the spine once, yielding a so-called biarc. In addition, all edges are intersection-free. We consider a variant called down-up monotone arc diagram where all biarcs must have the same monotone shape. This drawing style has applications in point-set-embeddability problems. We improve the best known upper bound for the number of biarcs in such arc diagrams and present a SAT formulation of the arc diagram drawing problem.

In the third and final part of the thesis, we shift our attention to beyond planar drawings which are motivated by the fact that visualization of nonplanar real-world graphs is necessary in many applications. In particular, we study RAC_k drawings, that is, drawings in which every edge is drawn as a polyline with at most k bends so that all intersections occur at right angles. First, we consider the density of graphs admitting RAC_1 drawings and give a new upper bound that we prove to be tight up to an additive constant. We also show that the graphs that admit simple RAC_1 drawings have slightly smaller edge density and provide a lower bound construction for this restricted scenario. Second, we investigate the area requirement of RAC drawings of dense graphs. Namely, we prove that every graph admits a RAC_3 drawing in cubic area and a RAC_8 drawing in quadratic area. In the case of p-partite graphs we even show how to achieve quadratic area RAC_3 drawings which we prove to not be possible for general graphs.

Finally, we conclude the thesis with a summary of our results and several interesting open research problems.

Zusammenfassung

Graphenzeichnen ist ein etabliertes Forschungsgebiet mit einer aktiven Forschungsgemeinschaft in der theoretischen Informatik, das die Einbettung von Graphen auf Oberflächen wie der euklidischen Ebene untersucht. Während traditionelle Ergebnisse aus der diskreten Mathematik stammen, haben sich Einbettungen von Graphen mit dem Aufkommen der Computertechnologie weiter verbreitet. Zwei wichtige Anwendungen sind das Design von Computerchips und verschiedene Diagrammtypen wie UML und BPMN, deren Entwicklung aufgrund der Allgegenwart von Computern in professionellen Umgebungen möglich war. Während die traditionellen Modelle, die sich aus diesen neueren Anwendungen ergeben, inzwischen gut untersucht sind, hat die Forschung begonnen, sich auf erweiterte Kontexte zu konzentrieren, deren Analyse technisch komplizierter ist. Eine der wichtigsten Forschungsrichtungen dieses Typs ist bekannt als Graph Drawing Beyond Planarity und untersucht Zeichnungen von Graphen, bei denen die Arten von Kantenkreuzungen eingeschränkt sind. In dieser Dissertation werden drei solche Modelle, die "beyond the beaten tracks" gehen, betrachtet.

Das wichtigste Zeichnungsmodell in Diagrammanwendungen sind orthogonale Zeichnungen, bei denen Kanten durch Polylinien dargestellt werden, deren Segmente achsenausgerichtet sind. Es ist nicht überraschend, dass es eine große Anzahl von Forschungsarbeiten gibt, die sich hauptsächlich mit planaren orthogonalen Zeichnungen beschäftigt. In dieser Thesis werden zwei Szenarien, die in der Literatur vorgeschlagen wurden und über dieses traditionelle Modell hinausgehen, untersucht: Im smooth-orthogonalen Zeichnungsmodell werden Knicke durch Kreisbögen ersetzt, während in oktilinearen Zeichnungen Segmente von Polylinien zusätzlich die Steigungen ± 1 aufweisen können. Im ersten Teil dieser Arbeit werden einige neue Ergebnisse für diese "beyond-orthogonalen" Zeichenstile gezeigt: Zunächst werden die Beziehungen zwischen den Klassen von Graphen, die smooth-orthogonale und oktilineare Zeichnungen zulassen, charakterisiert. Zweitens wird bewiesen, dass Techniken erforderlich sind, die von den traditionellen Methoden für orthogonale Diagrammzeichnungen abweichen, um smooth-orthogonale und oktilineare Zeichnungen mit minimaler Kurvenkomplexität zu berechnen. Drittens wird ein Zeichenalgorithmus angegeben, der mehrere wünschenswerte Eigenschaften garantiert. Anschließend werden orthogonale und smooth-orthogonale Zeichnungen 1-planarer Graphen, bei der Kanten höchstens einmal geschnitten werden dürfen, betrachtet. Hier werden effiziente Algorithmen zur Verfügung gestellt, die Zeichnungen berechnen, für deren Kurvenkomplexität in fast allen Szenarien gezeigt wird, dass sie worst-case-optimal sind.

Ein Graphenzeichnungstyp, der aus Anwendungen im VLSI-Chip-Design hervorgegangen ist, sind lineare Layouts. In einem linearen Layout sind die Knoten entlang einer Linie angeordnet, die Spine genannt wird, während die Kanten vollständig über der Spine gezeichnet werden. In einem Stack-Layout werden die Kanten zusätzlich eingefärbt, so dass sich keine zwei Kanten derselben Farbklasse

schneiden. Dieses Modell wurde insbesondere für planare Graphen eingehend untersucht. Das wichtigste Ergebnis in diesem Bereich ist, dass vier Farben für die Kanten jedes planaren Graphen ausreichen. Ein verwandtes Modell namens Queue-Layouts hat sich jedoch als wesentlich schwieriger zu analysieren erwiesen. In einem Queue-Layout sind die Kanten ebenfalls gefärbt, im Gegensatz zu Stack-Layouts dürfen sich jedoch Kanten derselben Farbklasse zwar schneiden, aber nicht schachteln. Über 25 Jahre lang war die Vermutung, dass eine konstante Anzahl von Farben für planare Graphen ausreicht, weder bewiesen noch widerlegt. Im zweiten Teil der Arbeit wird gezeigt, dass diese für planare Graphen begrenzten Grades in der Tat zutrifft. Dabei sei darauf hingewiesen, dass das Ergebnis inzwischen für alle planaren Graphen verallgemeinert worden ist. Es wird auch ein anderer Zeichenstil in Betracht gezogen, dessen Ausdrucksstärke jenseits der Möglichkeiten von Stack-Layouts liegt und Arc Diagram genannt wird. Im Gegensatz zu Stack-Layouts können die Kanten in einem Arc Diagram über und unter der Spine gezeichnet werden und die Spine sogar einmal schneiden, was einen sogenannten Biarc ergibt. Darüber hinaus sind alle Kanten kreuzungsfrei. Es wird eine Variante namens up-down-monotones Arc Diagram betrachtet, bei der alle Biarcs die gleiche monotone Form haben müssen. Dieser Zeichenstil findet Anwendung bei Point-Set-Embeddability-Problemen. Die beste bekannte Obergrenze für die Anzahl der Biarcs in solchen Arc Diagrams wird verbessert und eine SAT-Formulierung des Problems des Zeichnens von Arc Diagrams präsentiert.

Im dritten und letzten Teil der Arbeit wird der Fokus auf beyond-planare Zeichnungen gelegt, die dadurch motiviert sind, dass die Visualisierung nicht-planarer realer Graphen in vielen Anwendungen notwendig ist. Insbesondere werden RAC_k-Zeichnungen untersucht, d.h. Zeichnungen, in denen jede Kante als Polylinie mit höchstens k Knicken gezeichnet wird, so dass alle Kreuzungen im rechten Winkel auftreten. Zunächst wird die Kantendichte der Graphen, die RAC_1-Zeichnungen zulassen, betrachtet und eine neue Obergrenze angegeben, die bis auf eine additive Konstante optimal ist. Es wird auch gezeigt, dass die Graphen, die einfache RAC_1-Zeichnungen zulassen, eine etwas geringere Kantendichte aufweisen und es wird eine Konstruktion, die eine untere Schranke für die Dichte solcher Graphen darstellt, präsentiert. Zweitens wird der Flächenbedarf von RAC-Zeichnungen von dichten Graphen untersucht. Es wird bewiesen, dass jeder Graph eine RAC_3-Zeichnung in kubischer Fläche und eine RAC_8-Zeichnung in quadratischer Fläche zulässt. Im Falle von p-partiten Graphen wird sogar gezeigt, wie man RAC_3-Zeichnungen in quadratischer Fläche erreicht, was für allgemeine Graphen als nicht möglich bewiesen wird.

Schlußendlich wird die Arbeit mit einer Zusammenfassung der Ergebnisse und einigen interessanten offenen Forschungsproblemen abgeschlossen.

Acknowledgements

First and foremost, I want to thank my advisor Prof. Michael Kaufmann for sparking my interest in theoretical computer science, algorithms and graph drawing, for being a great advisor and co-author and for giving me the opportunities of writing this thesis, being part of his research group, and visiting several workshops and conferences in the past couple of years.

Moreover, I want to express my gratitude towards my other coauthors Patrizio Angelini, Fouli Argyriou, Michael Bekos, Till Bruckdorfer, Steve Chaplick, Sabine Cornelsen, Giordano Da Lozzo, Robert Ganian, Chris Geckeler, Martin Gronemann, Lukas Holländer, Fabian Klute, Stephen Kobourov, Myroslav Kryven, Beppe Liotta, Tamara Mchedlidze, Fabrizio Montecchiani, Martin Nöllenburg, Yoshio Okamoto, Tito Patrignani, Simon Poschenrieder, Chrysanthi Raftopoulou, Thomas Schneck, Amadäus Spallek, Jan Splett, Thomas Stüber, Torsten Ueckerdt, Sascha Wolff for the pleasant and fruitful collaborations. On this note, a special thanks has to go to Michalis for teaching me a lot about scientific work in the initial phase of my doctoral studies, and to Brillo Frati, Stephen Kobourov, Martin Nöllenburg and Antonis Symvonis for having me as a guest at their research groups for a couple of weeks.

My thanks also go to the past and present members of our work group Patrizio Angelini, Michael Bekos, Renate Hallmayer, Nik Heinsohn, Axel Kuckuk, Maxi Pfister, Lena Schlipf, Thomas Schneck and Alessandra Tappini not only for scientific discussions but also for the amazing work atmosphere.

For their delightful company especially also in the current weird times I want to thank my friends AJ, Alex, Anna, Anna, Chris, Eva, Jinglin, Marc, Natasha, Simon, Steffen, Stoyan, Tobi and everyone else I probably forgot now to mention by name (sorry! :().

Special thanks also go to the proofreaders Michael and Lena for the fast and valuable feedback.

Last but not least I want to thank my parents for their support.

Contents

Chapter 1

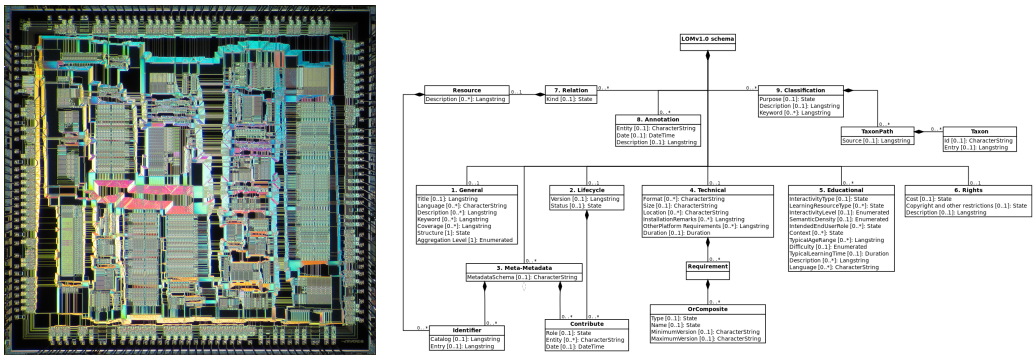
Introduction

Graph drawing deals with the embedding of graphs on surfaces such as the Euclidean plane. First results on topological graph embeddings date back at least to Euler [87, 88]. While initially a discipline of discrete maths, present-day graph drawing was vastly shaped by the ubiquity of computer science as new application domains became widespread: Graphs are a fundamental data structure in computer science and new technology such as VLSI design required special graph embeddings [125, 126, 127, 153, 156], diagrams like UML and BPMN diagrams[1] became increasingly more used in professional settings and motivated new fields of research [41, 62, 95] while graph visualizations in everyday applications such as metro maps that were traditionally drawn by hand could now be created with the support of a computer [110, 135]. Nowadays, graph drawing is an established discipline in theoretical computer science with an active research community and an annual conference.[2] While the traditional settings that are investigated in graph drawing are by now quite well understood, new topics of research have been identified and have become a main focus. In the following three paragraphs, we discuss exemplarily how three of the traditional models have matured and how new concepts extend beyond the classic framework. This thesis consists of three parts that each present several new results in one of these models which may be regarded as prototypical for current trends in graph drawing.

Beyond orthogonal drawings. The study of *orthogonal graph drawings* was motivated by applications in VLSI design and floor-planning [125, 127, 156]; see Fig. 1.1a. In an orthogonal drawing, edges are drawn as sequences of axis-aligned segments and attached to either the north, the south, the east or the west *port* of their endpoints which each may be used by only a single edge. A very important aspect in the evaluation of orthogonal drawings is the *curve complexity* which is the maximum number of segments of the drawing of any of the edges. In particular,

[1] See `https://www.omg.org/spec/UML` and `https://www.omg.org/spec/BPMN`.

[2] See `http://www.graphdrawing.org`.



(a) (b)

Figure 1.1: Typical applications of orthogonal graph drawings: (a) VLSI microprocessor[3], (b) UML diagram[4]. Observe the axis-aligned edge routing in both applications.

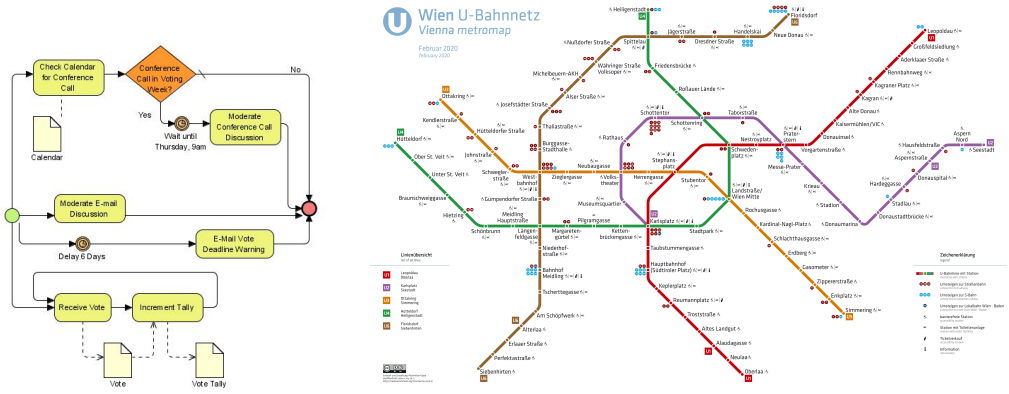
an orthogonal drawing of curve complexity k is referred to as an OC_k *drawing*.

Some of the most well-known works on orthogonal drawing proved that all planar graphs of maximum degree four admit a planar OC_3-drawing with the exception of the octahedral graph which admits a planar OC_4-drawing [43, 129] while all planar graphs of maximum degree three admit OC_2-drawings [117]. Note that this indicates that bounding the allowed curve complexity reduces the complexity of graphs that may be visualized.

In addition, it is noteworthy, that for maximum degree four planar graphs, it is NP-hard to find a drawing with minimum curve complexity [100]. On the other hand, the problem becomes polynomial time solvable if an embedding is given [149]. More precisely, in the latter scenario, one typically first computes an *orthogonal representation* that defines the angles between edges around a vertex and the number and types of bends along an edge. Based on this representation, segments are assigned a length in a subsequent step. This approach is well-known as the *topology-shape-metrics approach* [149] since one computes the orthogonal representation from a topological embedding that defines a shape, before computing the metrics of the resulting drawing. A very important aspect here is, that every orthogonal representation in which the angles around each face of length k

[3]Pauli Rautakorpi: "Die shot of Tandem VLSI VF4723 microprocessor designed by Tandem Computers for CLX 800 series fault tolerant computers and manufactured by VLSI Technology. Chip has both Tandem and VLSI logos and VLSI, 9404TV 341321, VF4723-0001, 31093B00 CPU, TANDEM USA as markings", 21 August 2018. Taken from `https://commons.wikimedia.org/wiki/File:VLSI_Tandem_CLX_800_CPU_die.jpg`. Published under CC-BY 3.0, see `https://creativecommons.org/licenses/by/3.0/legalcode` for the license information.

[4]EleGall: "An UML diagram of IEEE1484.12.1 Learning Object Metadata (LOM) base schema", 7 August 2011. Taken from `https://commons.wikimedia.org/wiki/File:LOM_base_schema.svg` Published under CC-BY SA 3.0, see `https://creativecommons.org/licenses/by-sa/3.0/legalcode` for the license information.



(a) **(b)**

Figure 1.2: (a) In BPMN diagrams, it is common practice to replace bends with circular arcs[5]. (b) Metromaps are typically drawn with octilinear edges[6].

sum up to the valid amount of $(k-2)\pi$ also admits a realizing orthogonal drawing.

The orthogonal graph drawing model restricts graphs to have maximum degree four due to the port restriction. Especially in the context of UML and BPMN diagrams, this has been relaxed so to accommodate multiple edges at the same side of a vertex which also allows graphs of maximum degree larger than four to be visualized; see e.g. Fig. 1.1b. This model is known as the *Kandinsky model*. Due to its practical applications it is unsurprising, that a large body of literature has focused on Kandinsky drawings [36, 41, 44, 45, 62, 95, 96]. Another limitation in the practical application of orthogonal drawings is that, traditionally, research has mainly focused on planar graphs. For the scenario where intersections cannot be avoided, some algorithms exist that however cannot guarantee in which patterns these intersections occur, see e.g. [43, 96].

Over the years, in some application domains, drawing types have evolved that may be regarded as extensions of the orthogonal graph drawing model. For instance, in BPMN diagrams, it is common practice to replace bends with circular arcs; see Fig. 1.2a. As a result, for maximum degree four graphs optimal angular resolution can be maintained while also realizing each edge with a smooth curve. In the literature, this concept has been introduced as *smooth orthogonal drawings*, in which every edge consists of a sequence of axis-aligned straight-line segments and circular arcs such that two consecutive segments have an overlapping tangent at their shared point. In addition, the port restriction is adopted from orthogo-

[5]Figure taken from `https://commons.wikimedia.org/wiki/File:BPMN-CollectVotes.jpg`.

[6]HerrMay: "Metromap of Vienna in December 2019", 18 February 2020. Taken from `https://commons.wikimedia.org/wiki/File:U-Bahnnetz_Wien_2019.png` Published under CC-BY SA 3.0, see `https://creativecommons.org/licenses/by-sa/3.0/legalcode` for the license information.

nal drawings. In contrast to common practice in many of the applications, it is even possible to entirely draw edges with circular arcs, which may even lead to a reduction in curve complexity. Similar to orthogonal drawings, we call a smooth orthogonal drawing with curve complexity k an SC_k *drawing*.

The main results of the literature state that all planar graphs of maximum degree four admit an SC_2 drawing [9] while planar graphs of maximum degree three [29] admit an SC_1 drawing. The principles of the Kandinsky drawing model were also applied to smooth orthogonal drawings, achieving drawings of curve complexity two [31]. Moreover, it is known that every maximum degree four graph admits an SC_1 drawing [29], but as it was the case with orthogonal drawings, no guarantees for the patterns of intersection can be made. Finally, it is noteworthy that it is conjectured that testing if a planar graph of maximum degree four admits a planar SC_1 drawing is NP-hard [9].

Another extension of orthogonal drawings can be found in metro maps, where the number of slopes used by segments of edges is also limited. Usually, metro maps use the two axis-aligned slopes and the slopes 1 and -1; see Fig. 1.2b. Drawings in which the slopes of segments are restricted in this fashion are known as *octilinear drawings*. Octilinear drawings are also often used in the typical applications of orthogonal drawings to allow for a larger maximum degree without adopting the Kandinsky model. Again, the curve complexity is an important quality measure and an octilinear drawing of curve complexity k is known as an $8C_k$ *drawing*.

All planar graphs of maximum degree eight admit a planar $8C_3$-drawing [121] while if the maximum degree is five a planar $8C_2$-drawing is possible [28]. Moreover, planar graphs of maximum degree three always admit a planar $8C_1$-drawing [69, 116]. Deciding if a maximum degree eight planar graph admits a planar $8C_1$-drawing is however NP-hard [135]. This raises the question which is the lowest maximum degree for which NP-hardness holds. Results on octilinear Kandinsky drawings so far have been limited mostly to a restricted subcase called *Sloginsky drawings* [33, 36].

In Part I, we will investigate several properties of smooth orthogonal and octilinear drawings. In addition, we will also reconsider the classic orthogonal drawing model and show new results for non-planar graphs in which intersections are restricted. Both of these perspectives extend beyond the classic planar orthogonal graph drawings.

Beyond stack layouts. Another drawing type that became popular due to its application in VLSI design [57, 58] is known as *stack layout*. A stack layout is a *linear layout*. In a linear layout, all vertices are restricted to a line S called *spine* while edges are drawn in disjoint halfplanes delimited by S. The occurence of vertices along S induces a total order $\prec$. We then say that two edges (u, v) and (u', v') are *intersecting*, if $u \prec u' \prec v \prec v'$. Also we say that (u, v) *nests* (u', v'), if $u \prec u' \prec v' \prec v$. In a stack layout no two edges on the same halfplane may

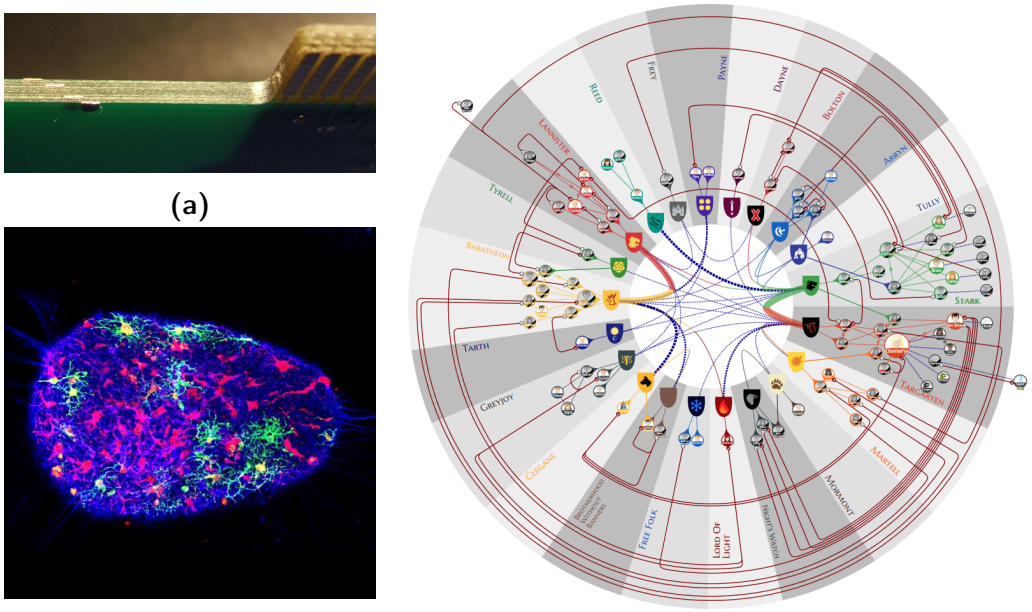


Figure 1.3: Applications of linear layouts: (a) Multi layer printed circuit board[7], (b) 3D graph drawing[8], and, (c) circular layout[9].

intersect, hence the subdrawing on each halfplane is planar. A natural application are multi layer printed circuit boards: Typically, the input and output pins (vertices) are located at one side of the circuit board while in the different layers connections (edges) are routed; see Fig. 1.3a.

The most important parameter of a graph class $\mathcal{G}$ related to stack layouts is the *stack number* which is the minimum number s such that each graph belonging to $\mathcal{G}$ admits a stack layout where the edges use at most s halfplanes. The graphs with stack number one and two are exactly the outerplanar and subhamiltonian graphs, respectively [40]. Since the recognition of subhamiltonian graphs is NP-hard, computing the stack number is NP-hard as well. On a positive side, it is known that the stack number of planar graphs is four [30, 161]. We point out that there are also results dealing with nonplanar graphs in the literature, see e.g. [75].

[7]Christoph Kappel: "Multi layer printed circuit board with copper gold ISA", 02 April 2006. Taken from `https://commons.wikimedia.org/wiki/File:Multilayer_pcb.jpg`. Published under CC-BY SA 3.0, see `https://creativecommons.org/licenses/by-sa/3.0/legalcode` for the license information.

[8]Eltobgy: "Three-dimensional brain graph", 8 December 2017. Taken from `https://commons.wikimedia.org/wiki/File:Three-dimensional_brain_graph.jpg`. Published under CC-BY SA 4.0, see `https://creativecommons.org/licenses/by-sa/4.0/legalcode` for the license information.

[9]Evmorfia Argyriou, Michael Baur, Anne Eberle and Armin Gufler: Winning contribution to the Graph Drawing Contest 2018. Taken from `http://mozart.diei.unipg.it/gdcontest/contest2018/results.html`.

In addition to stack layouts, the related drawing style known as *queue layouts* has been investigated in the literature. A queue layout is also a linear layout, but in contrast to stack layouts edges assigned to the same halfplane may intersect but not nest. We point out that it may appear as if allowing intersections on a halfplane would allow for graphs to be realized which are more complex than those realizable with stack layouts with the same number of halfplanes. However, the graphs that may be realized in one halfplane are the *arched-level planar graphs* [108] which have the same density as outerplanar graphs and are a subfamily of planar graphs. Despite the potentially many intersections on each halfplane, queue layouts are related to intersection-free three dimensional graph visualizations [68, 78] and have been motivated by three-dimensional circuit layouts [126]. Over the decades, three dimensional graph drawings have become more widespread, especially in applications that deal with three-dimensional data or simply have the need to visualize large complex networks, see also Fig. 1.3b.

The queue number is defined analogously to the stack number. More precisely, the queue number of a graph class $\mathcal{G}$ is the minimum q such that each graph belonging to $\mathcal{G}$ admits a queue layout where the edges use at most q halfplanes. Based on the queue number, the relation to three-dimensional graph drawings can be defined more clearly, namely, graph classes of bounded queue number admit an intersection-free three-dimensional drawing in linear volume [68, 78]. At the start of this thesis, bounded queue numbers have been known only for not necessarily planar graphs for which some other graph parameter is bounded [23, 78, 81, 107, 158]. In this context, it is also noteworthy that bounding the degree is not bounding the queue number in general [160]. On the other hand, similar results were known only for more limited subfamilies of planar graphs [8, 98, 143]. Moreover, the best known upper bound for the queue number of planar graphs was $\mathcal{O}(\log n)$ [23]. In Chapter 5, we will show that the queue number of bounded degree planar graphs is bounded. In the meantime, it was shown, that this result extends to all planar graphs [76].

Another application of stack layouts arises in circular layouts, where vertices are restricted to a circle; see e.g. Fig. 1.3c. In fact, by glueing together both ends of the spine, such a layout is easily obtained [65]. Moreover, if the stack layout only uses two halfplanes, one of them can be projected to the outside of the circle while the second can be projected to the inside. If the stack number however is at least three, this approach cannot be easily applied as projecting two pages to the same side of the circle may introduce arbitrary intersections. One strategy to cope with this is to use *arc diagrams* instead of stack layouts. In an arc diagram, one restricts the number of halfplanes to two but allows edges to be composed of sequences of segments such that each segment is located on either halfplane and starts and ends on the spine. If an arc diagram is mapped to a circle to create a circular layout, an edge composed of more than one segment is partially drawn inside and outside of the circle. Arc diagrams also find applications in point set embeddability problems [15, 89, 130].

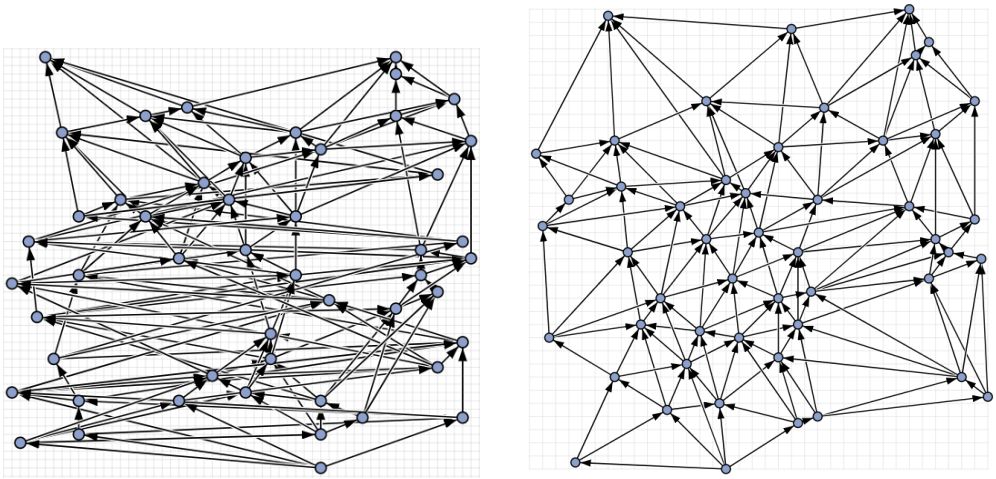
It is known that every planar graph admits an arc diagram in which each edge is either drawn as a *proper arc* (i.e., with one segment) or as a *biarc* (i.e., with two segments) [120]. Biarcs have been investigated for additional properties. Namely, biarcs can be *monotone*, if their intersection with the spine occurs between their corresponding endpoints. Moreover, a monotone biarc can be *down-up*, if its left segment regarding the spine appears on one halfplane that is defined as the "bottom" halfplane while its second segment is located on the other halfplane called "top". This down-up property is used in several works that build upon arc diagrams [89, 130]. In fact, every planar graph admits a so-called *down-up monotone arc diagram*, in which every edge is either a proper arc or a down-up monotone biarc [67]. While biarcs are required in general, one may hope to use as few of them as possible. It was shown, that every planar graph admits an arc diagram with $\lfloor (n-3)/2 \rfloor$ not necessarily monotone biarcs and a down-up monotone arc diagram with $n-4$ biarcs while in general $\lfloor (n-8)/3 \rfloor$ biarcs may be required [51].

In Part II, we first show that the queue number of bounded degree planar graphs is bounded. This complements similar results that are known for stack layouts. Second, we investigate the number of biarcs in down-up monotone arc diagrams that are stack layouts on two half-planes where the spine may be intersected. Both of these models go beyond the classic study of stack layouts of planar graphs.

Beyond planar drawings. Planar graphs have been the traditional focus of graph drawing research. Hence, it comes at no surprise that planar graphs are quite well understood and that efficient planar drawing algorithms exist for popular drawing styles such as straight-line drawings [60], orthogonal drawings [149] or upward drawings [102]. The literature for nonplanar drawings initially considered properties such as the number of required intersections [7, 124] while algorithmic works often did not discuss the intersection patterns created [43].

In contrast, many graphs arising in real-world applications are not planar. In addition, it has been shown that drawings are easier to interpret if the angles formed by intersecting edges are rather large [113, 115]; see also Fig. 1.4. As a result, in the past decade, a new research direction called *graph drawing beyond planarity* became a main focus of graph drawing [72]. In this line of research, drawing styles are considered in which the patterns formed by intersections are restricted. The two probably most well-known beyond planar drawing styles are 1-*planar drawings*, in which every edge is intersected at most once, and *right angle crossing drawings* (or *RAC drawing*, for short), in which each pair of intersecting edges is perpendicular. Note that we also say that a graph is 1-*planar* or *RAC* if it admits a 1-planar drawing or a RAC drawing, respectively.

Most work on 1-planar drawings deals with straight-line drawings. It is known that not every 1-planar graph admits a 1-planar straight-line drawing [85, 154]. However, every triconnected 1-planar graph admits a straight-line 1-planar drawing



(a) **(b)**

Figure 1.4: Two drawings of the same graph[10]. Drawing (b) is clearly easier to read than drawing (a). Observe that in drawing (b) every edge is intersected at most once while the angle between two intersecting edges is fairly large; these two properties cannot be found in drawing (a).

except for at most one edge on the outer face [10]. This and the fact that testing 1–planarity is NP-hard [101] stand in contrast to similar results on planar graphs. It is also known that deciding if a given 1-planar drawing can be straightened can be done in linear time [112].

For RAC drawings, an important parameter is the curve complexity. It is easy to see that every graph admits a RAC drawing for sufficiently large curve complexity, however, this would come at the expense of potentially many bends per edge. Hence, the literature considers *RAC_k drawings*, which are RAC drawings with at most k bends per edge for small values of k. Depending on the number of bends per edge, the density of the graphs, that can be realized, differs. The density of graphs that admit a RAC_0 drawing have at most $4n-10$ edges which is also tightly achieved by an infinite class of graphs [71] which is in fact 1-planar [84]. Another similarity to 1-planarity is that testing whether a graph is RAC_0 is NP-hard [17], even in some restricted settings [14, 24]. RAC_1 and RAC_2 graphs both also have a linear number of edges [19]: The best known upper bounds are $6.5n-13$ and $74.2n$ for the edge density of RAC_1 and RAC_2 graphs, respectively, while there exist lower bound constructions achieving $4.5n-\mathcal{O}(\sqrt{n})$ and $7.83-\mathcal{O}(\sqrt{n})$, respectively. Finally, every graph admits RAC_3 drawing in $\mathcal{O}(n^4)$ area [71]. Subsequently, this area bound was improved to $\mathcal{O}(n^3)$ for RAC_4 drawings [66] and recently to $\mathcal{O}(n^{2.75})$

[10]Both drawings are taken from `http://mozart.diei.unipg.it/gdcontest/contest2019/results.html#manual` and were created in the scope of the Graph Drawing Contest 2019. Subfigure (a) shows the input for problem 5 created by the contest committee, subfigure (b) the best manual submission due to team *Dinosaurs*.

for RAC_6 drawings [142].

In Part III, we prove new results for polyline RAC drawings, both in terms of edge density and area. These results are prototypical for the research in the field of graph drawing beyond planarity.

Overview of the thesis. The remainder of this thesis is structured as follows. We first provide basic definitions and formally introduce the drawing styles discussed in this thesis in Chapter 2. In addition, we also discuss the related literature in more careful detail in this chapter.

In Part I, we investigate some new directions in the analysis of the beyond orthogonal graph drawing models discussed above. Namely, in Chapter 3, we first investigate the relationship between graphs admitting SC_1 and graphs admitting $8C_1$ drawings, respectively. Then, we prove that the topology-shape-metrics approach cannot be easily adopted to smooth orthogonal and octilinear drawings despite their relationships to orthogonal drawings. In particular, we show that deciding whether a given shape can be realized is NP-hard. Finally, we also provide efficient Kandinsky drawing algorithms for both drawing styles. Then, in Chapter 4, we begin the study of orthogonal and smooth orthogonal drawings with restricted intersection patterns, namely, we show how to draw 1-planar and outer-1-planar graphs in the orthogonal and smooth orthogonal drawing style while maintaining the embedding. In addition, except for the smooth orthogonal outer-1-planar setting, we show that the resulting drawings have worst-case optimal curve complexity.

In Part II, we show new results for queue layouts and arc diagrams. Namely, in Chapter 5 we prove that the queue number of planar graphs of bounded degree is bounded. This partially affirms a long-standing conjecture by Heath, Leighton and Rosenberg [107] and is the first result standing in strict contrast to similar results for nonplanar graphs [160]. We point out that subsequently to our publication [25], it was shown that the queue number of planar graphs is bounded in general [76]. In Chapter 6, we then investigate arc diagrams and prove that every planar graph admits a down-up monotone arc diagram with $15/16n - \mathcal{O}(1)$ biarcs. While the improvement upon the previous best known algorithm [51] is not very large, we point out that our algorithm is much more technically elaborate. Finally, we present a SAT formulation for computing arc diagrams.

In Part III, we consider polyline RAC drawings. More precisely, in Chapter 7, we continue the study of RAC_1 graphs and achieve a tight density bound of $5.5n - \mathcal{O}(1)$. In addition, we show, that there are RAC_1 graphs that admit no *simple* RAC_1 drawing, that is, in any of their RAC_1 drawings some pairs of edges share more than one point. We do so, by providing a density upper bound of $5.4n - 10.8$ for the class of graphs with simple RAC_1 drawings. In addition, we demonstrate that there are simple RAC_1 drawings of infinitely many graphs with $5n - 10$ edges which still improves upon the best known previous lower bound. In Chapter 8, we

then shift our attention to RAC_k drawings with $k \geq 3$. We improve the results for the area of RAC_3 drawings and show that cubic area is sufficient while quadratic area is not. In addition, we close the gap between upper and lower bound of the area of RAC drawings by showing that every graph admits a RAC_8 drawing in quadratic area. Moreover, we show that the required curve complexity can be reduced to three bends per edge if the input graph is p-partite.

We then conclude the thesis in Chapter 9 by summarizing our results and presenting some open problems. We point out that the results appearing in this thesis have been previously published in [12, 18, 25, 26, 52, 94]. Most of the figures in this thesis have been adopted from those publications and were transformed in a consistent style. Other works coauthored by the author of the thesis during the thesis period have been published in [11, 16, 27, 53, 54, 93]; we provide an annotated list of those works in Appendix A.

Chapter 2

Preliminaries and Related Work

In this chapter, we provide definitions and preliminary results that will be used in the subsequent chapters. Further, we provide an overview over previous work related to the results of this thesis. Technical details of previous results that our proof techniques build upon can be found in the corresponding chapters instead. As this thesis deals with graph drawing, we first establish several definitions and properties related to graph theory in Section 2.1 before we proceed to introduce several commonly used principles and techniques from graph drawing in Section 2.2. Finally, we discuss the drawing styles we focus on in this thesis in Section 2.3.

2.1 Graph Theoretic Foundations

We first give formal definitions of the fundamental concepts used in this thesis in Section 2.1.1. Then, in Section 2.1.2, we show how graphs can be categorized based on structural properties before introducing several graph families occuring in the remaining chapters in Section 2.1.3. Unless specified otherwise, the contents of this section can also be found in [74, Ch.1]. For a more in-depth introduction to graph theory, we refer the reader to [74, 159].

2.1.1 Graphs

In the context of this thesis, the term *(simple) graph* refers to a tuple $G = (V, E)$ where V is called the set of *vertices* and $E \subseteq V \times V$ is referred to as the set of *edges*. Graphs are commonly used to model relational data and form a fundamental data structure in many applications of computer science. For the sake of brevity, we will denote the number of vertices by n and the number of edges by m.

Let $e = (u, v)$ be an edge. We refer to u and v as *endpoints* (or *endvertices*) of edge e and we say that e is incident to u (and v). We call edge $e = (u, v) \in E$ *directed*, if and only if $(v, u) \notin E$. Otherwise we call $e = (u, v)$ *undirected* and

also denote it by $\{u, v\}$ and (v, u). In addition, if $e = (u, v)$ is directed, we call u *source* (or *initial* vertex) of e and v *target* (or *terminal* vertex) of e. Moreover, we say that G is *directed* if all its edges are directed and we say that G is *undirected* if all its edges are undirected. For a directed graph G, we call vertex v a *source* of G if there is no edge $e \in E$ such that v is target of e and we call v a *sink* of G if there is no edge $e \in E$ such that v is source of e [159, Ch.7]. Unless otherwise specified we consider undirected graphs. Let G be an undirected graph and G' be a graph obtained by choosing a source and a target for every edge of G. Then, we call G' *orientation* of G.

Let v be a vertex. We call u *neighbor* of v if $(u, v) \in E$. Further, we denote by $N(v) = \{u \in V | (v, u) \in E\}$ the *neighborhood* of v. Further, we denote by $deg(v)$ the *degree* of vertex v, that is, the number of edges incident to v. For a subgraph S of G, we denote by $deg_S(v)$ the degree of vertex v in S. In addition, we say that G has *maximum (vertex) degree* Δ, or for the sake of brevity $deg(G) = \Delta$, if there is a vertex $v \in V$ with $deg(v) = \Delta$ but no vertex $v' \in V$ with $deg(v) > \Delta$. If all vertices in G have degree exactly k, we call G *k-regular*.

Let $S \subseteq V$ be a subset of the vertices of G and let $E_S = \{(u, v) \in E | u \in S \wedge v \in S\}$ be the set of edges with both endpoints in S. We call $G[S] = (S, E_S)$ the *subgraph of G induced by S*.

2.1.2 Connectivity

Let $G = (V, E)$ be a graph. Let $P = (v_1, \ldots, v_k)$ be a sequence of vertices in V. We call P a *path* in G if and only if $\forall i : 1 \leq i < k : (v_i, v_{i+1}) \in E$. We call G *k-connected* if and only if for any pair of vertices $u, v \in V$ there exist at least k paths in G between u and v which are vertex-disjoint except for u and v. Note that 1-connected graphs are also known as *(simply) connected* graphs whereas 2- and 3-connected graphs are also known as *biconnected* and *triconnected* graphs, respectively [138].

The following observations may provide a helpful intuition for connectivity: In a connected graph which is not also biconnected, there exists a single vertex, called *cut vertex*, whose removal will separate the graph. Moreover, in a biconnected graph which is not also triconnected, there exists a pair of vertices, called *separation pair*, whose removal separates the graph. Finally, if a triconnected graph is not 4-connected, it contains a triple of vertices whose removal would separate the graph. Moreover, if G is k-connected, by definition, $deg(G) \geq k$.

In a connected graph, the *graph-theoretic distance* $dist(u, v)$ between two vertices u and v is the number of edges in a shortest path connecting them.

2.1.3 Special Families of Graphs

Let $G = (V, E)$ be a graph. Let $X = (v_1, \ldots, v_k)$ be a sequence of k vertices of V. We call X a *cycle* in G if and only if $(v_i, v_{(i+1) \bmod k}) \in E$ for $1 \leq i \leq k$ and we call X a *path* in G if and only if $(v_i, v_{(i+1)}) \in E$ for $1 \leq i < k$. We call G a *cycle* (or *path*, respectively) if and only if G entirely consist of a cycle (or path, respectively. Further, we call G a *tree* if and only if G contains no cycles and G is connected. Note that each path is also a tree. Let $T = (V, E)$ be a tree. We call each vertex of degree one in T a *leaf*. Moreover, if T consists of a path P and leafs which are connected to one vertex of P each, we call T a *caterpillar* and P the *spine* of T [105]. Trees can be *rooted* at a special vertex r called *root*, in such a case, we assume that all edges are directed such that their target has a larger graph-theoretic distance from the root than their source. Let (u, v) be a directed edge in a rooted tree. We call u *parent* of v and v *child* of u [59, Ch.10]. A tree in which each vertex has at most k children is called k-*ary*; a k-ary tree has maximum degree $k + 1$ [59, Ch.10]. A k-ary tree is called *complete* if each inner vertex has exactly k children. We say that a rooted tree T has *height* h if the graph-theoretic distance between root r to any of the other vertices of T is at most h. A *spanning tree* T of a graph $G = (V, E)$ is a subgraph of G that is a tree and includes all vertices of V. We call a spanning tree T a *BFS-tree*[1] with root $r \in V$ if the graph-theoretic distance between every vertex v and r is the same in T and G [59, Ch.22]. A BFS-tree can be computed in time $\mathcal{O}(n+m)$ [59, Ch.22].

If each connected component of G is a tree we call G a *forest*. A forest that consists of paths of length at most one is called a *matching*. If in a matching, every vertex is incident to an edge, we call the matching *perfect*. Note that perfect matchings are also known as 1-*factors* in the literature [74, Ch.2]. If G consists of a cycle $C = (v_1, \ldots, v_k)$, a vertex c and edges (c, v_i) for $1 \leq i \leq k$, we call G a $(k+1)$-*wheel*, c the *center* of G and C the *rim* of G [159, Ch.2].

If the vertices of G can be partitioned into p sets $V_1, \ldots, V_p$ such that $G[V_i]$ contains no edges for $1 \leq i \leq p$, we call G p-*partite*. Let n_i denote the number of vertices in subset V_i. We call the graph $K_{n_1,\ldots,n_p}$ a *complete p-partite graph* if each vertex in V_i is connected to each vertex in V_j for $1 \leq i < j \leq p$. If $n_1 = n_2 = \ldots = n_p$, we also write $K_{(n_p)_p}$ instead of $K_{n_p,\ldots,n_p}$. Moreover, we call the graph $K_n = (V, V \times V)$ with $|V| = n$ the *complete graph on n vertices*.

2.2 Graph Drawing Basics

Here, we first provide several basic definitions in Section 2.2.1 and desirable properties of graph layouts in Section 2.2.2. Then, we will introduce techniques commonly used in graph drawing in Section 2.2.3 which we will apply in the subsequent

[1] The name derives from the fact that a BFS-tree can be computed with breadth-first-search.

chapters. The contents of Section 2.2.1 and 2.2.2 can, unless noted otherwise, for instance also be found in [157] and [133, Ch.1], respectively, we also refer the reader to [63, 119, 133, 150] for a more in-depth introduction into graph drawing.

2.2.1 Basic Definitions

A *drawing* Γ of a graph $G = (V, E)$ is a function that maps each $v \in V$ to a point $\Gamma(v) \in \mathbb{R}^2$ and each edge $(u, v) \in E$ to a simple open Jordan curve $\Gamma((u, v))$ with endpoints $\Gamma(u)$ and $\Gamma(v)$. We refer to $\Gamma(v)$ (or $\Gamma(e)$) as the *representation* of vertex $v \in V$ (or edge $e \in E$, respectively) in drawing Γ. Conversely, we say that G *admits drawing* Γ. If all vertices are located on integer coordinates, we call Γ a *grid* drawing. If in Γ the representations of all edges are sequences of straight-line segments, we also call Γ a *polyline* drawing whereas we call Γ *straight-line* if every edge is represented by one straight-line segment. We say that two edges $e, e' \in E$ *intersect* if their representations $\Gamma(e)$ and $\Gamma(e')$ share at least one common point which is not a shared endpoint. A point in the plane that is shared by at least two edge representations is referred to as an *intersection* if it is not a shared endpoint. We call Γ *planar* if there exists no pair of edges $e, e' \in E$ such that representations $\Gamma(e)$ and $\Gamma(e')$ intersect. Further, we say that G is a *planar graph* if and only if G admits a planar drawing. Let $S \subseteq V$. We call the drawing $\Gamma[S]$ of graph $G[S]$ contained in Γ the *subdrawing of G induced by S*.

A planar drawing Γ of a graph G partitions $\mathbb{R}^2$ into topologically connected regions. We refer to these regions as *faces* and to the one unbounded face as *outer* (or *external*) *face*. If in Γ all vertices are incident to the outer face, we call Γ an *outerplanar* drawing. We say that a face f is *incident* to a vertex v, if v appears on the boundary of f. We say that a face has *degree* $deg(f)$, if $deg(f)$ vertices are incident to f and we say that a face has *length* $\ell(f)$ if the cyclic order of vertices along the boundary of f repeats after $\ell(f)$ vertices. Note that it may occur that $deg(f) \neq \ell(f)$. More precisely, we also define a face f by the cyclic order $(v_1, \ldots, v_{\ell(f)})$ of vertices along its boundary which also uniquely defines the corresponding region bounded in Γ.

We say that two planar drawings Γ_1 and Γ_2 are *topologically equivalent*, if there exists a bijection ϕ between the set of faces F_1 of Γ_1 and the set of faces F_2 of Γ_2 such that for every face $f_1 \in F_1$ it holds that f_1 and $\phi(f_2)$ are bounded by the same cyclic order of vertices. A class of topologically equivalent drawings of a graph G is referred to as *(topological) planar embedding* of G. Let $\mathcal{E}$ be a planar embedding of a graph G. For two drawings $\Gamma_1, \Gamma_2 \in \mathcal{E}$ it holds that for every vertex $v \in V$, the cyclic order of edges around v is the same in Γ_1 and Γ_2. Let $S \subseteq V$. Similarly to induced subgraphs, we call the embedding $\mathcal{E}[S]$ of graph $G[S]$ induced by the restrictions of the cyclic orders in $\mathcal{E}$ to S the *subembedding of G induced by S*. We refer to the tuple $(G, \mathcal{E})$ as a *plane graph*, for the sake of brevity, we may also write that G is a plane graph which implicitly assumes that a suitable

embedding $\mathcal{E}$ is part of the input. The terms outerplanar embedding and outerplane graph are defined analogously. Note that the term embedding can be equivalently defined by an equivalence function that considers the cyclic order of edges around each vertex, in this case we call the equivalence classes *combinatorial embeddings*. This distinction can be found in the literature for two reasons: First, sometimes a specified outer face is considered to be part of a topological embedding but not of a combinatorial embedding. Second, combinatorial embeddings generalize naturally to nonplanar drawings since only the cyclic orders of edges around each vertex are considered whereas faces are not well-defined in nonplanar drawings; we will discuss nonplanar drawings further in Section 2.3.2. It is worth noting that triconnected planar graphs have only two embeddings (where one can be obtained from the other by a mirroring operation).

Let F denote the set of faces of an embedded planar graph G. The *dual graph* $G^* = (V^*, E^*)$ of G has a vertex for every face of G, i.e., $V^* = F$. Moreover, two vertices $f_1, f_2 \in V^*$ are connected by edge $(f_1, f_2) \in E^*$ if and only if the f_1 and f_2 share an edge in G. For triconnected planar graphs, it holds that the dual of the dual graph is the initial graph, i.e., $(G^*)^* = G$. In particular, for outerplanar graphs, the *weak dual graph* $G^\star$ is considered [92]. Let f_0 be the outer face of the embedded graph G. Then, the weak dual graph $G^\star$ is the subgraph of G^* induced by all vertices but f_0, i.e., $G^\star = G^*[F \setminus \{f_0\}]$. Most notably, in a outerplane graph G, $G^\star$ is a tree.

Maximal planar graphs are planar graphs which cannot be augmented by any edge without destroying planarity. In particular, every maximal planar graph is triconnected and a triangulation of the plane, i.e., in all of its two embeddings it partitions the plane into triangular faces. Moreover, it is always possible to augment a planar graph to a maximal planar graph by adding more edges. The resulting triangulations however may depend on the chosen embedding that was triangulated. Since this augmentation is always possible, we will assume that graphs are maximal planar where appropriate.

In a triconnected planar graph which is not 4-connected, the triples of vertices whose removal separates the graph may be assumed to form induced 3-cycles, called *separating triangles*. Since separating triangles form difficult configurations for many algorithms in graph drawing, *Kleetopes* are an important graph class often appearing in upper and lower bound constructions. A *Kleetope* can be obtained from a planar graph G by placing a vertex v_f in each face f of G and connecting v_f to all vertices on the boundary of f [103]. Note that often Kleetopes based on triangulations are considered as every new vertex is placed inside a separating triangle in such a Kleetope.

2.2.2 Aesthetic Criteria

Several *aesthetic criteria* have been proposed in the literature to evaluate the quality of a graph drawing [63, 119, 133]. Here, we only introduce those which will be important in the scope of this thesis. In general, for a given graph drawing algorithm, we will measure its aesthetic quality by giving functions in the number of vertices (or edges) as bounds for the aesthetic criteria that will be achieved when applying the algorithm to a valid input graph.

The *area* and the *aspect ratio* of a graph drawing Γ are usually defined by a minimally sized axis-parallel bounding rectangle $R \subseteq \mathbb{R}^2$ encapsulating Γ. The *area* of Γ is defined as the area of R whereas the *aspect ratio* is equal to the ratio of the length of the long side of R divided by the length of the short side of R. In the case of grid drawings, R is well defined, otherwise, the shortest distance between any two vertices in Γ can be considered as the unit distance to define R. We are mostly interested in small area drawings while we intend to maintain a small aspect ratio.

In non-planar drawings, we may be interested in the *crossing number* and the *number of intersections per edge* in order to measure to which extent planarity is deviated. The *crossing number* is a global parameter that measures the total number of intersections. The *number of intersections per edge*, on the other hand, counts the maximum number of intersections on any edge, and hence can provide more information on the local structure of the graph drawing.

In drawing styles where the representations of edges are sequences of geometric primitives, such as straight-line segments and circular arc segments, the *curve complexity* is an important aesthetic criterion. If in a drawing Γ the representation of each edge consists of at most k such primitives, we say that Γ has *curve complexity* k. A low curve complexity is a desirable property of a graph drawing.

The intersection points of two consecutive primitives are referred to as *bends*. The *smoothness* of a drawing, measured by the minimum angle formed by the tangents of the two primitives meeting at any bend, also plays an important role in the readability of the drawing [114]. In this thesis, we will consider three special cases: First, we will consider *smooth* drawings, that is, drawings in which at each bend the tangents of both primitives overlap. Second, we will consider the *orthogonal* case, that is, the case where the angle between such tangents is always right. The third special case will be the *octilinear* case where the angles between the tangents at each bend is one of $\{\frac{\pi}{4}, \frac{\pi}{2}, \frac{3\pi}{2}\}$.

Finally, we will also consider the *angular* and the *crossing resolution* of a graph drawing Γ. The *angular resolution* is equal to the smallest angle formed by any pair of edges at a common endpoint. A large angular resolution indicates that there is few clutter around the representation of vertices. The *crossing resolution*, on the other hand, measures the smallest angle formed by any pair of intersecting edges at their intersection. A large crossing resolution is desirable as otherwise it

may be difficult to follow the trajectory of intersecting edges.

2.2.3 Vertex Orderings and Shift Method

In this section, we outline two important vertex orderings, namely, st-orderings for biconnected graphs [128] and the canonical ordering [60] for maximal planar graphs. Also, we present the *shift method*, which is the most well known application of the canonical ordering [60].

Let $\pi = (v_1, \ldots, v_n)$ be a permutation of the vertices of a biconnected graph $G = (V, E)$ and let $s = v_1$ and $t = v_n$. Then, we call π an *st-ordering* if and only if for $2 \leq i \leq n-1$, there exists $1 \leq j < i < k \leq n$ such that $(v_j, v_i), (v_i, v_k) \in E$ [128]. Further, for vertex v_i we say that i is the *st-number* of v_i. Most notably, st-orderings are used to employ a nice orientation of a graph. In particular, directing all edges from the vertex with lower st-number to the one with larger st-number yields an orientation of G such that s is the only source and t the only sink of G. An st-ordering exists for all biconnected graphs and each pair of vertices s and t used as source and sink, respectively [128]. In fact, for planar graphs with a fixed embedding $\mathcal{E}$, selecting as s and t two vertices on the same face yields an st-ordering that allows for a *planar upward drawing*, that is, a planar drawing that preserves $\mathcal{E}$ in which vertices are placed at a y-coordinate equal to their corresponding st-number while edges are drawn monotone in y-direction [145].

Let $\pi = (v_1, \ldots, v_n)$ be a permutation of the vertices of a maximal planar graph $G = (V, E)$. For $1 \leq k \leq n$, let $V_k := \{v_1, \ldots, v_k\}$ and $G_k = G[V_k]$. Further, for a fixed embedding $\mathcal{E}$ of G with outer face (v_1, v_2, v_n), let C_k denote the outer face of $\mathcal{E}[V_k]$. We call π a *canonical ordering* if and only if for $2 \leq k \leq n$

(i) G_k is biconnected and internally triangulated

(ii) the neighbors of v_k belonging to G_{k-1} appear consecutively along C_{k-1}, and,

(iii) unless $k = n$, v_k has at least one neighbor v_ℓ with $\ell > k$.

A canonical ordering of a maximal planar graph can be computed in linear time; in fact, a well-known linear time algorithm [117] even generalizes to triconnected planar graphs. For biconnected planar graphs several variants have been proposed in the literature [102, 104, 106]; however, not all algorithms using the canonical order can be easily adapted to these variants. We also point out that the canonical ordering can be seen as a special st-ordering for $s = v_1$ and $t = v_n$.

Moreover, a canonical ordering of a triangulation G is equivalent to a *Schnyder realizer* of G [147]. Let (v_1, v_2, v_3) be one of the faces of G. A Schnyder realizer is a set of three rooted trees $\{T_1, T_2, T_3\}$ such that

(i) every edge $e \in E$ belongs to exactly one of T_1, T_2 and T_3,

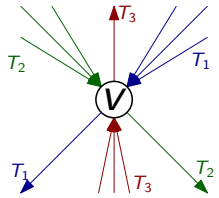


Figure 2.1: Ordering of edges around a vertex in a Schnyder realizer $\{T_1, T_2, T_3\}$.

(ii) G is exactly the union of T_1, T_2 and T_3,

(iii) T_1 is a spanning tree of G, T_2 a spanning tree of G without v_1 and T_3 a spanning tree of G without v_1 and v_2,

(iv) the root of T_1 is v_1, the root of T_2 is v_2 and the root of T_3 is v_3, and,

(v) for every vertex $v \in V$, the counter-clockwise order of edges around v is as follows: edge to the parent in T_1, edges from the children in T_3, edge to the parent in T_2, edges from the children in T_1, edge to the parent in T_3, edges from the children in T_2; see Fig. 2.1. Note that the three trees are oriented from leaf to root.

In the literature, Schnyder realizers are often colored in a specific way. In particular, T_1 is usually colored blue, while T_2 and T_3 are colored green and red, respectively [90]. From a given canonical ordering, a Schnyder realizer can be obtained as follows: For vertex v_k consider the consecutive sequence of neighbors $(w_\ell, \ldots, w_r)$ along C_k. Then, we define $(v_k, w_\ell) \in T_1$, $(v_k, w_r) \in T_2$ and $(v_k, w_i) \in T_3$ for all $\ell < i < r$. Further, we select v_1 to be the root of T_1, v_2 to be the root of T_2 and v_n to be the root of T_3. We will use such a *Schnyder coloring* of a graph regarding a canonical ordering to classify edges in the analysis of algorithms using the canonical ordering.

Next, we outline a linear time algorithm, known as *shift-method*, that produces straight-line drawings for maximal planar graphs [60]. The algorithm incrementally inserts vertices according to a canonical ordering π and produces a drawing Γ_k for the subgraph G_k induced by k already inserted vertices that satisfies the following invariants:

I.1 Γ_k is straight-line and planar and all vertices are located on integer coordinates.

I.2 Γ_k fits into an area of width $2k - 4$ and height $k - 2$.

I.3 All edges on C_k have slope 1 or -1 except for edge (v_1, v_2) which has slope 0. For an illustration, refer to Fig. 2.2a.

In addition, each vertex v on C_k will be associated with a so-called *shift-set* $S(v) \subset V_k$ such that each vertex in V_k is contained in exactly one shift-set. Based on the shift-sets there exists the following additional invariant:

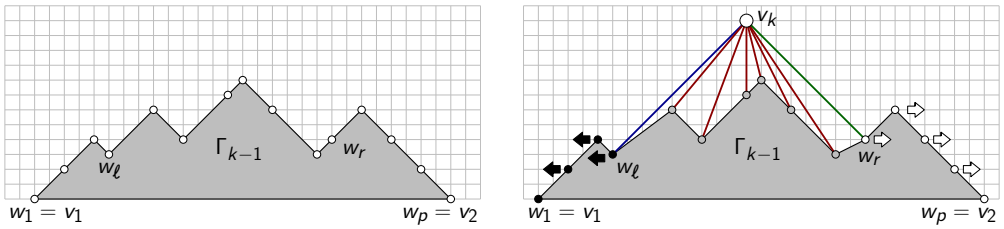


Figure 2.2: (a) Illustration of Invariant 3 maintained by the shift-method, and, (b) illustration of the shifting operation and insertion of the new vertex in an iteration of the shift-method.

I.4 Let $(u, v) \in G_k$ such that u and v appear in different shift-sets. Then the slope of $\Gamma_k(u, v)$ is in the interval $[-1, 1]$. Moreover, in a Schnyder coloring, edge (u, v) is blue or green.

As an initial drawing, G_3 is realized as $\Gamma_3(v_1) = (0, 0)$, $\Gamma_3(v_2) = (2, 0)$ and $\Gamma_3(v_3) = (1, 1)$ and $S(v_i) = \{v_i\}$ for $1 \leq i \leq 3$. For $4 \leq k \leq n$, assume that we have a straight-line drawing Γ_{k-1} and shift-sets for the vertices on C_{k-1}. Let $(w_1 = v_1, \ldots, w_p = v_2)$ denote the vertices along C_{k-1} from left to right and let w_ℓ and w_r denote the leftmost and rightmost neighbor of v_k on C_{k-1}, respectively. Observe that by definition of the canonical ordering, v_k is also incident to w_i for $\ell < i < r$ and that v_k is the last neighbor of w_i to be inserted. In order to avoid overlappings of edges, first, for $1 \leq i \leq \ell$, all vertices in $S(w_i)$ are translated one unit to the left, while for $r \leq i \leq p$, all vertices in $S(w_i)$ are translated one unit to the right. Note that this operation only affects the representations of edges between shift-sets $S(w_{\ell-1})$ and $S(w_\ell)$ as well as for the representations of edges between $S(w_r)$ and $S(w_{r+1})$; all these edges are green or blue and the remaining edge representations are simply shifted. After this shifting operation, v_k is placed on the intersection point of the line of slope 1 passing through w_ℓ and the line of slope -1 passing through w_r; refer to Fig. 2.2b. Finally, the shift set of v_k is defined as $S(v_k) = \bigcup_{i=\ell}^{r} S(w_i) \cup \{v_k\}$.

2.3 Graph Drawing Styles

In this section, we define the graph drawing models investigated in this thesis and give an overview over related work on the specific drawing styles.

2.3.1 Orthogonal Drawings and Extensions

Orthogonal Drawings. In an *orthogonal drawing* Γ of a graph $G = (V, E)$, each vertex is represented as a point in the plane whereas each edge e is represented

as a sequence of horizontal and vertical line segments $\Gamma(e)$. In addition, each vertex v is associated with four distinct *ports* called north, west, south and east. If the segment of the representation of edge e that is attached to v has v as its bottom (right, top, left) endpoint, we say that e *uses* v's north (west, south, east; respectively) port. As a further restriction to orthogonal drawings, each of the ports of a vertex can be used only by one edge. Hence, the maximum degree of a graph admitting an orthogonal drawing is at most four. For an example of an orthogonal drawing refer to Fig. 2.3b.

Orthogonal graph drawing dates back to applications in VLSI design and floorplanning which where studied in the early 1980's [125, 127, 156]. Nowadays, the most important applications include UML or BPMN diagrams. Hence, it comes at no surprise that the orthogonal graph drawing model has been deeply studied. For an overview of well-established techniques we refer the reader to [63, 119, 150] while we point the reader to [43, 95, 129, 149] for some efficient algorithmic solutions providing high quality drawings.

By definition, orthogonal drawings achieve perfect angular resolution for graphs of maximum degree four while they also avoid acute angles at the bends of edges. These properties contribute to the good readability of orthogonal drawings. Additionally, orthogonal graph drawing algorithms usually also try to optimize the area and the curve complexity of the resulting layout. For the sake of brevity, we will call an orthogonal drawing of curve complexity k an OC_k*-drawing*. It is known that all planar graphs of maximum degree four admit a planar OC_3-drawing with the exception of the octahedral graph which admits a planar OC_4-drawing [43, 129]; these drawings require $O(n) \times O(n)$ area. Moreover, if the maximum degree of a planar graph is three, it admits a planar OC_2-drawing [117].

It is also known that each connected graph of maximum degree four admits a (not necessarily planar) OC_3-drawing with no guarantees regarding the number of intersections per edge [43].

The Topology-Shape-Metrics framework. Finding an orthogonal drawing with a minimum number of bends over all embeddings of a maximum degree four planar graph has been shown to be NP-hard [100]. For plane graphs of maximum degree four, it can however be solved in polynomial time by reduction to two min-cost flow problems [149]. More precisely, given the embedding (that is, a *topology*), the solution of the first network flow defines an *orthogonal representation*, which defines the angles between any two edges incident to the same vertex and the bends encountered on each edge. The second network flow problem is based on a feasible representation (or *shape*) and the flow algorithm computes how long each edge should be (that is, the *metrics* of the drawing) in order to find a drawing that *realizes* the given representation. We say that an orthogonal drawing Γ *realizes* an orthogonal representation $\mathcal{R}$, if the angles between any pair of edges incident to the same vertex in Γ is identical to the angle prescribed in $\mathcal{R}$

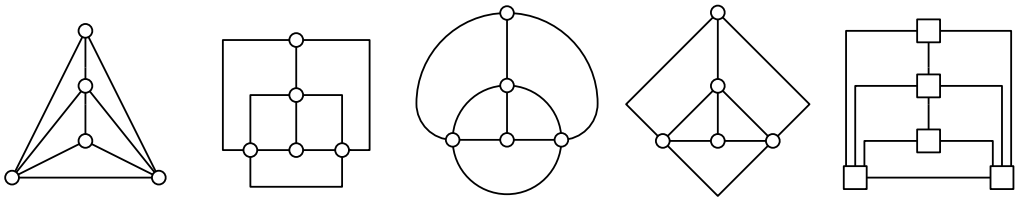


Figure 2.3: Five drawings of the same graph using different drawing styles: (a) Straight-line drawing, (b) Orthogonal drawing, (c) Smooth orthogonal drawing, (d) Octilinear drawing, and, (e) Kandinsky drawing.

and if in Γ, the bends occuring on each edge are the ones described by $\mathcal{R}$. This framework of graph drawing is also known as the *topology-shape-metrics* approach as the drawing is produced in three steps (where the first *topology* step is to decide for an embedding). It is worth pointing out that each feasible orthogonal representation also can be realized. This makes it so that each of the algorithms used for solving one of the three steps of the framework are interchangeable without influencing the other steps.

In the following, we discuss two extensions of the orthogonal graph drawing model which each allow one more type of geometric primitives in the representations of edges, namely *smooth orthogonal drawings* [31] which also allow circular arcs and *octilinear drawings* which also allow diagonal straight-line segments. Afterwards, we discuss *Kandinsky drawings* [95] which generalize orthogonal drawings to graphs of larger maximum degree.

Smooth Orthogonal Drawings. A *smooth orthogonal drawing* Γ of a graph $G = (V, E)$ maps each vertex $v \in V$ to a point $\Gamma(v) \in \mathbb{R}^2$ and each edge $e \in E$ to a sequence of straight-line segments and quarter, half and three-quarter circular arcs such that the tangents of consecutive segments overlap at their shared bend. In addition, the port restriction of the orthogonal model is also used in smooth orthogonal drawings. As in orthogonal drawings, smooth orthogonal drawings can only visualize graphs of maximum degree at most four. For an example of a smooth orthogonal drawing refer to Fig. 2.3c. Observe that the orthogonal drawing of the same graph in Fig. 2.3b has curve complexity three which is in fact necessary while there even exists a (less symmetric) smooth orthogonal drawing of curve complexity one.

Smooth orthogonal drawings have been proposed by Bekos et al. [31] as a drawing style which combines the easy readability and rigidity of the well established orthogonal drawing style with the artistic appeal and smoothness of *Lombardi drawings*[2] [82, 86]. As a result, smooth orthogonal drawings achieve optimal

[2]In a *Lombardi drawing*, edges are represented by circular arc segments. In addition, all angles between consecutive pairs of edges around a vertex have the same size.

angular resolution for graphs of maximum degree four while also realizing each edge with a smooth curve. Similar to the orthogonal graph drawing model, other important parameters to evaluate the quality of smooth orthogonal drawings are the area and the curve complexity of the layout. We will refer to a smooth orthogonal drawing of curve complexity k as an SC_k *drawing*.

It is known that planar SC_1-drawings always exist for maximum degree three planar graphs [29] and for outerplanar graphs of maximum degree four [9] both requiring superpolynomial area. In contrast, there exist maximum degree four planar graphs which do not admit a planar SC_1-drawing [31]. On the other hand, all planar graphs of maximum degree four admit a planar SC_2-drawing. The required area may however be exponential for graphs of maximum degree four, while it is cubic for graphs of maximum degree three [9]. The problem of deciding if a planar graph of maximum degree four admits a planar SC_1-drawing is open and has been conjectured to be NP-hard [9].

In addition, each graph of maximum degree four admits an SC_1-drawing with no guarantees regarding the number of intersections per edge [29].

Octilinear Drawings. In an *octilinear drawing*, each vertex is represented by a point in the plane whereas each edge is drawn as a sequence of straight-line segments at slopes $\{0, -1, 1, \infty\}$. In contrast to orthogonal drawings and smooth orthogonal drawings, each vertex is assigned eight ports (north, north-west, west, south-west, south, south-east, east, and north-east) and as a consequence octilinear drawings can realize graphs of maximum degree at most eight. More intuitively speaking, octilinear drawings add two additional slopes for segments to the orthogonal graph drawing model. For an example octilinear drawing refer to Fig. 2.3d. Observe that the curve complexity of the example drawing is two whereas the orthogonal drawing of the same graph in Fig. 2.3b has curve complexity three. Moreover, when comparing the octilinear layout to the smooth orthogonal one in Fig. 2.3c, we can observe that in both drawings all bends and all vertices have the same position. We will investigate this observation further in Section 3.1.

The most common applications of octilinear drawings include metro-maps and map schematization [110, 135, 134, 148]. In addition, octilinear drawings generalize the concept of orthogonal drawings for graphs with maximum degree at most eight while also a variant called *slanted orthogonal drawings*[3] has been investigated in the literature [32, 35] for representing nonplanar graphs of maximum degree four. The most important aesthetic criteria for octilinear drawings include area, curve complexity and smoothness of edges since diagonal and axis-aligned segments can meet at their shared bend either at $\frac{\pi}{4}$ or $\frac{3\pi}{4}$. As with orthogonal and smooth orthogonal drawings we will refer to an octilinear drawing of curve complexity k as an $8C_k$ *drawing*.

[3] In *slanted orthogonal drawings*, vertices are incident to axis-aligned segments while intersection occur on diagonal segments. This faciliates distinction between vertices and intersections.

It has been shown that all planar graphs of maximum degree eight admit a planar $8C_3$-drawing [121]. Moreover, for planar graphs of maximum degree five it is possible to compute a planar $8C_2$-drawing which even fits in polynomial area if the maximum degree is at most four [28]. In addition, planar graphs of maximum degree three always admit a planar $8C_1$-drawing [69, 116]. The problem of deciding whether a planar graph of maximum degree eight admits a planar $8C_1$-drawing has been shown to be NP-hard [135] while it is not known if NP-hardness is still true for planar graphs of lower maximum degree. In particular, this question is especially interesting for planar graphs of maximum degree four since as mentioned earlier planar graphs of maximum degree three are always positive instances whereas there exist planar graphs of maximum degree four without a planar $8C_1$-drawing [34].

Kandinsky Drawings. In a *Kandinsky drawing* Γ of a graph $G = (V, E)$ [41, 62, 95, 96], each vertex $v \in V$ is represented by an axis-aligned square $\Gamma(v)$ and each edge $(u, v) \in E$ is represented by a sequence of axis-aligned straight-line segments $\Gamma((u, v))$ whose endpoints touch $\Gamma(u)$ and $\Gamma(v)$ at exactly one point. In the classic model, the squares representing vertices have uniform side length Δ where Δ denotes the maximum degree of G so that the side to which each edge attaches can be chosen arbitrarily. Moreover, in the case of grid drawings, vertices are centered on a *coarse grid* whose grid lines are sufficiently far apart to avoid vertices to overlap (i.e. at $\Omega(\Delta)$ distance) whereas bends of edges are located on a *fine grid* (usually fine grid lines have unit distance). For an example of a Kandinsky drawing refer to Fig. 2.3e.

Kandinsky drawings relax the strict port constraints of the orthogonal graph drawing style to allow drawings for graphs of arbitrary degree while also it may be possible to reduce the curve complexity for graphs of maximum degree four (compare for instance the drawings in Figs. 2.3b and 2.3e). They find wide usage in VLSI design and UML and BPMN diagrams where usually vertices are realized by two-dimensional objects due to application constraints. Due to their practical applications, Kandinsky drawings have received much attention in research over the decades, see for instance [36, 44, 45].

The Kandinsky drawing model allows for natural extensions to smooth orthogonal and octilinear drawings which so far have only received little attention in the literature. A topological book embedding (as defined in Section 2.3.3) can be used to produce a planar SC_2-Kandinsky drawing for any planar graph in linear time [31]. The resulting drawings have quadratic area and all vertices are located on a line while edges are represented by half circular arcs or as a sequence of two half circular arcs (one above and one below the line). Further, it was shown that not all planar graphs admit an SC_1-Kandinsky drawing [29]. In the octilinear setting the most notable application of the Kandinsky style are the so-called *Sloginsky* drawings [33, 36] which generalize slanted orthogonal drawings to higher maximum degree. As in slanted orthogonal drawings, the additional slopes are used for

segments on which intersections occur to make them easier distinguishable from vertices.

2.3.2 Beyond Planar Graph Drawings

A currently popular research direction in graph drawing is the study of so-called *beyond planar* graphs, that is, classes of graphs which extend planarity by admitting drawings in which the configurations formed by intersections are restricted. This line of research was motivated by experimental studies verifying that a large crossing resolution indeed yields well readable drawings [113, 115]. As a result, one of the earliest studied beyond planar graph classes is the class of *Right-Angle-Crossing* graphs, or *RAC* graphs for short, that is, the class of graphs, admitting some drawing where all pairs of intersecting edges intersect at a right angle. The most commonly used restriction to achieve meaningful RAC drawings is to limit the curve complexity, which is also supported by experiments as an important aesthetic criterion [140, 141]. With a fixed curve complexity, it might become increasingly more difficult to realize drawings with many intersections per edge such that all intersections are at right angles. This observation and the notion that few intersections per edge do not impede readability too much motivate the continuation of the study of so-called *k-planar graphs*, that is, graphs in which every edge is intersected at most k times. The class of k-planar graphs had received some attention in research (typically with slightly different scope) as early as the 1960's [144]. Additionally, k-planar graphs prove to have nice structural properties which can be used in many algorithms. Consequently, k-planar graphs have become another well-studied class of graphs, especially for the case $k = 1$; see [123] for a survey of results. In the scope of this thesis, we only consider the classes of RAC and 1-planar graphs, however, we point out that more classes have been investigated in the literature. We refer the interested reader to [72] for a recent survey.

1-planar graphs. 1-planar graphs were already introduced by Ringel in 1965 when studying simultaneous colorings of a planar graph G and its dual G^* [144]. Most importantly, straight-line drawings have been studied for 1-planar graphs in the literature. In contrast to planar graphs, not every 1-planar graph admits a 1-planar straight-line drawing [85, 154]. However, every triconnected 1-planar graph admits a straight-line 1-planar drawing except for at most one edge on the outer face [10]. Moreover, it can be decided in linear time, if a given 1-planar drawing can be straightened [112]. However, it is NP-hard to compute a 1-planar embedding [101]; in fact, the result was recently extended to the computation of k-planar embeddings [155].

A subclass of 1-planar graphs that we will consider in this thesis is the class of *outer-1-planar* graphs. An outer-1-planar graph admits a 1-planar drawing where all vertices are on the boundary of one topologically connected region. It

is known that such graphs are planar and can be recognized in linear time [20, 109]. Other subclasses that have been considered in the literature include *IC-planar (independent crossings)* [48] and *NIC-planar graphs (nearly independent crossings)* [21], in which, pairs of intersecting edges share no or at most one vertex, respectively.

RAC graphs. In a RAC drawing of a graph all pairs of intersecting edges intersect at a right angle. If the curve complexity is unbounded, this can be easily achieved by bending each edge arbitrarily close to its intersections. As a result, with very few exceptions [54], research has focused on so-called *RAC_k drawings*, that is, polyline RAC drawings in which each edge has at most k bends, or, curve complexity $k + 1$, which were introduced in [71]. Similarly to planar graphs, we call a graph *RAC_k* if it admits a $\mathrm{RAC_k}$ drawing. In the following, we provide an overview over previous results.

$\mathrm{RAC_0}$ or straight-line RAC graphs with n vertices have at most $4n - 10$ edges which is also tightly achieved by an infinite class of graphs [71]. Moreover, the $\mathrm{RAC_0}$ graphs that have exactly $4n - 10$ edges are 1-planar [84]. The relationship to 1-planar graphs has been further studied and it is known that the subclass of IC-planar graphs always admits a $\mathrm{RAC_0}$ drawing [48] while the subclass of NIC-planar graphs does not [21]. Moreover, outer-1-planar graphs admit $\mathrm{RAC_0}$ drawings [61]. The problem of testing whether a graph is $\mathrm{RAC_0}$ is NP-hard [17], even in the restricted case where the output drawing must be 1-planar [24] or upward [14]. In contrast, the complete bipartite graphs admitting $\mathrm{RAC_0}$ drawings are fully characterized [70] and the biconnected graphs that admit a $\mathrm{RAC_0}$ drawing with the vertices restricted to two parallel lines can be recognized in linear time [64]. Outer-$\mathrm{RAC_0}$ graphs, that is, graphs admitting a $\mathrm{RAC_0}$ drawing with the vertices restricted to the unbounded region of the drawing have also been investigated [111].

Already the first paper on RAC drawings [71] established that $\mathrm{RAC_1}$ and $\mathrm{RAC_2}$ graphs have a subquadratic *edge density*, that is, they have a subquadratic number of edges. However, it was subsequently shown that indeed both have only linearly many edges [19]: The best known upper bounds are $6.5n - 13$ and $74.2n$ for the edge density of $\mathrm{RAC_1}$ and $\mathrm{RAC_2}$ graphs, respectively, while there exist lower bound constructions achieving $4.5n - \mathcal{O}(\sqrt{n})$ and $7.83 - \mathcal{O}(\sqrt{n})$, respectively. Interestingly, all graphs of maximum degree three admit a $\mathrm{RAC_1}$ drawing while all graphs of maximum degree six admit a $\mathrm{RAC_2}$ drawing [14]. Further, there exists an embedding-preserving algorithm that can compute a $\mathrm{RAC_1}$ drawing of a 1-planar graph in super-polynomial area [24]. Polynomial area on the other hand can be achieved in $\mathrm{RAC_2}$ drawings or when the 1-planar graph is in fact NIC-planar [55]. In addition, if the maximum degree is three, a $\mathrm{RAC_2}$ drawing, in which each edge segment has one of three possible slopes, can be computed [122].

It is known, that any graph with n vertices admits a $\mathrm{RAC_3}$ drawing in $\mathcal{O}(n^4)$ area [71]. This area bound was later improved to $\mathcal{O}(n^3)$ for $\mathrm{RAC_4}$ drawings [66] and

more recently to $\mathcal{O}(n^{2.75})$ for RAC_6 drawings [142]. In addition, a relaxation called *large angle crossing* drawings, or *LAC* drawings for short, has been considered. In an α-*LAC drawing*, pairs of intersecting edges intersect at an angle of $\alpha = \frac{\pi}{2} - \varepsilon$ for some small deviation angle $\varepsilon > 0$. It has been shown that every graph admits an α-LAC drawing with one bend per edge in $\mathcal{O}(n^2 \cot^2 \frac{\varepsilon}{2})$ [66]. We emphasize that this construction requires that ε is strictly greater than 0 while the multiplicative factor of $\cot^2 \frac{\varepsilon}{2}$ albeit being a constant can be prohibitively large for small values of ε and moderate values of n. In the cited studies, vertices and bends of edges are placed on an underlying integer grid while the positions of intersections are implicitly defined by the endpoints of the intersecting segment. If also intersections must be located on the grid, $\Omega(n^4)$ area is required since in any drawing of the complete graph $\Omega(n^4)$ intersections are required by the crossing lemma [7, 124]. Hence, in this version of the problem, an area optimal algorithm is known [71] while in the normal setting, the lower bound is $\Omega(n^2)$ area.

Commonly used techniques. Beyond planar drawings can be grouped into equivalence classes to which we refer as *beyond planar embeddings*. For instance, a 1-*planar embedding* specifies the cyclic order of edges incident to each vertex and which pairs of edges form intersections. In contrast, a *k-planar embedding* for $k \geq 2$ specifies in addition the sequence of intersections along each edge. As typically the problem of computing a beyond planar embedding is *NP*-hard, algorithms producing beyond planar drawings usually assume that a valid embedding is provided as an input.

Given an embedding $\mathcal{E}$ of a beyond planar graph G that specifies the sequence of intersections along each edge, a *planarization* G_p of G can be computed by replacing every intersection with a so-called *dummy vertex* of degree four [50]. As $\mathcal{E}$ specifies the sequence of intersections along each edge, it also defines how dummy vertices have to be connected with each other or with real vertices. The planarization G_p can then be used as input for planar drawing algorithms or as an auxiliary graph in the process of drawing G directly.

Another important tool is the so-called *crossing lemma* which allows us to give a bound on the required number of intersections based on the density of a graph [7, 124]. More precisely, for a graph with m edges and n vertices, it holds that in any drawing there are $\Omega(m^3/n^2)$ intersections.

Finally, in some cases drawings are required to be *simple*, that is, the pair of representations of each pair of edges share at most one point which can be either an endpoint or an intersection. Straight-line drawings are simple by definition, however, requiring simplicity can impose further restrictions beyond the forbidden pattern of intersecting edge. For instance, the class of quasiplanar graphs[4] has a maximum edge density of $7n - \mathcal{O}(1)$ while the class of simple quasiplanar graphs has a maximum edge density of $6.5n - 13$ [5]; both bounds being tight.

[4] A quasiplanar graph admits a drawing in which no triple of edges pairwise intersect.

Figure 2.4: Four different relationships between edges (u, v) and (u', v') in a linear layout: (a) Independent, (b) Dependent, (c) Nesting, and, (d) Intersecting.

2.3.3 Linear Layouts

A special type of graph drawings are *linear layouts* in which the drawings of all vertices are restricted to a line called *spine*. In this thesis, we mainly consider two types of linear layouts, namely, *queue layouts* and *arc diagrams*. Since both of these linear layout types are also closely related to so-called *stack layouts*, we will also introduce those in this section.

Stack and Queue Layouts. Consider a total ordering $\prec$ of the vertices of a graph G. When arranging the vertices according to $\prec$ on the spine, we may observe four different relationships between edges (u, v) and (u', v'):

(i) If $u \prec v \prec u' \prec v'$, we say that (u, v) and (u', v') are *independent*; see Fig. 2.4a.

(ii) If (u, v) and (u', v') have a common endpoint, we say that (u, v) and (u', v') are *dependent*; see Fig. 2.4b.

(iii) If $u \prec u' \prec v' \prec v$, we say that (u, v) *nests* (u', v'); see Fig. 2.4c.

(iv) If $u \prec u' \prec v \prec v'$, we say that (u, v) and (u', v') *intersect*; see Fig. 2.4d.

Based on these relationships, we can define *stack* and *queue layouts*. A *k-page stack layout*[5] of a graph G consists of a total ordering $\prec$ of its vertices and an assignment of its edges to k color classes called *pages* such that no two edges (u, v) and (u', v') of the same color *intersect*. Conversely, a *k-page queue layout* of a graph G consists of a total ordering $\prec$ of its vertices and an assignment of its edges to k color classes called *pages* such that no two edges (u, v) and (u', v') of the same color *nest*. The names "stack" and "queue" derive from the fact, that when traversing the vertices from the left end to the right end of the spine, edges of the same color can be pushed or enqueued when processing the left endpoint while they can be popped or dequeued when processing the right endpoint. We say that a graph G has *stack number* $sn(G)$ or *queue number* $qn(G)$ if G admits a $sn(G)$-page stack layout but no $(sn(G) - 1)$-page stack layout or $qn(G)$-page queue layout but no $(qn(G) - 1)$-page queue layout, respectively. Also we say that

[5]Stack layouts are also known as *book embeddings* in the literature.

a graph class $\mathcal{G}$ has *stack number* $sn(\mathcal{G})$ or *queue number* $qn(\mathcal{G})$ if $sn(\mathcal{G})$ or $qn(\mathcal{G})$ is the minimum number such that for every $G \in \mathcal{G}$, it holds that $sn(G) \leq sn(\mathcal{G})$ or $qn(G) \leq qn(\mathcal{G})$, respectively.

Both stack and queue layouts have been successfully used in VLSI design [57, 58, 126] and sorting [153]. In addition, queue layouts have been applied in scheduling applications [42] and despite allowing intersections on the same page have a surprising connection to graph drawing: Namely, graph classes of bounded queue number admit an intersection-free three-dimensional drawing in linear volume [68, 78].

For visualization purposes, the edges assigned to one page may be embedded on a halfplane delimited by the spine. While many intersections can occur on the same page of queue layouts, a single page of a stack layout is indeed outerplanar since the spine is on the boundary of the associated halfplane; in fact the outerplanar graphs are exactly the graphs with stack number 1 [40]. By using the halfplanes above and below the spine for embedding both pages, a 2-page stack layout can be easily converted into a planar drawing. However, only *subhamiltonian* planar graphs admit a 2-page stack layout, that is, planar graphs that can be augmented with planar edges so to contain a Hamiltonian cycle [40]. As a result, computing the stack number of a graph is NP-hard. Otherwise, for planar graphs, up to four pages may be required [30, 161]. In addition, k-planar graphs have stack number $\mathcal{O}(\log n)$ [75] while in general, the stack number can be as large as $\lceil n/2 \rceil$ [40] for nonplanar graphs.

In contrast to stack layouts, outerplanar graphs have queue number 2. Surprisingly enough, the graphs of queue number one are a subclass of planar graphs called *arched-level planar* [108] which even have the same density as outerplanar graphs. Since testing for arched-level planarity is NP-complete [107], also computing the queue number of a graph is a difficult computational problem. Similar to stack layouts, the complete graph K_n has queue number $\lfloor n/2 \rfloor$ [108]. Sublinear upperbounds are known for graphs with $m \in o(n^2)$ edges [107] and minor-closed graph families [78]. Constant upper bounds are known for not necessarily planar graphs of bounded tree- and pathwidth [78, 158], bounded tracknumber [81], bounded bandwidth [107] and bounded layered pathwidth [23].

At the start of the work on this thesis, results for queue numbers of planar graphs in the literature were rather limited. It was known that subclasses of planar graphs have constant queue number, namely, Halin graphs [98], series-parallel graphs [143] and planar 3-trees [8]. The best known upper bound for the queue number of planar graphs was $\mathcal{O}(\log n)$ [23] while the best known lower bound was four for a family of planar 3-trees [8]. For planar graphs on the other hand, Heath, Leighton and Rosenberg had conjectured that the queue number was bounded [107]. During this thesis, we positively answered the conjecture for planar graphs of bounded degree [25] which stands in contrast to general bounded degree graphs [160]; we also discuss this result in Chapter 5. Subsequently, the conjecture was positively answered for all planar graphs [76].

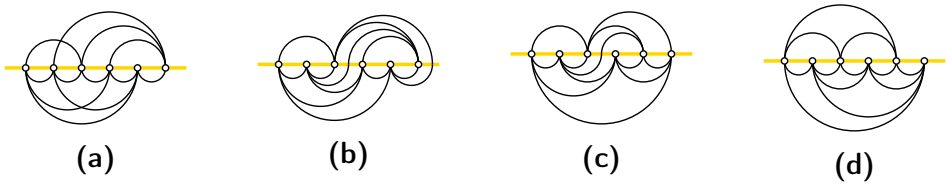


Figure 2.5: Four arc diagrams of the same graph: (a) General arc diagram with intersections, (b) Plane arc diagram, (c) Plane monotone arc diagram, (d) Plane proper arc diagram (or 2-page stack layout).

Arc Diagrams. As previously discussed, only subhamiltonian planar graphs admit a 2-page stack layout while general planar graphs require up to four half-planes for realizing their edges. On the other hand, even 3-page stack layouts cannot be easily embedded into the plane; drawing two half-planes on top of each other may introduce many intersections. This is in contrast to the possibility to embed a 2-page stack layout intersection-free in the plane by drawing one half-plane above and one below the spine.

In order to retain the clarity of stack layouts when restricting the drawing to two half-planes, it is however possible to draw some edges as sequences of segments which alternate between both half-planes. Such layouts are known as *arc diagrams*; see Fig. 2.5. While the first works on arc diagrams considered the number of intersections [1, 132, 146], due to the previously discussed motivation, *plane* arc diagrams also have been considered, that is, arc diagrams, in which edges do not intersect [51, 67]; see Fig. 2.5b. Two special types of edge drawings in arc diagrams are *proper arcs* and *biarcs* which consist of exactly one and two segments, respectively. Additionally, the drawings of edges can be *monotone* with respect to the spine. An arc diagram in which all edges are monotone is called a *monotone arc diagram*; see Fig. 2.5c. In addition, an arc diagram in which all edges are proper arcs is called *proper arc diagram*; see Fig. 2.5d.

It is known that all planar graphs admit a planar arc diagram in which each edge is represented by a proper arc or a biarc [120]. This result originates from the study of *point set embeddability*, that is, the problem of defining a point set S whose size is polynomially bounded by n such that all planar graphs on n vertices admit a planar drawing with vertices located on one of the points in S where edges are drawn in some restricted way (specifically here polylines with at most 2 bends). Moreover, a biarc can be *down-up*, that is, it is monotone such that the spine can be oriented so that the left segment of the biarc is below and the right segment of the biarc is above the spine [67]. We call an arc diagram in which all biarcs are down-up *down-up monotone*.

While in general biarcs are required, it is also possible to guarantee that a majority of edges is drawn as proper arcs; the best known upper and lower bounds on the required number of biarcs are the following [51]: Every planar graph admits an arc diagram with $\lfloor (n-3)/2 \rfloor$ not necessarily monotone biarcs and a down-up

monotone arc diagram with $n - 4$ biarcs. On the other hand, there are planar graphs that require at least $\lfloor (n-8)/3 \rfloor$ biarcs in any arc diagram.

Planar arc diagrams are also used as an intermediate tool for related graph drawing problems. For instance, they can be easily transformed to a *circular layout* [65], that is, a polyline drawing where vertices are restricted to a circle; see also Fig. 2.6a. In addition, they found applications in point set embeddability problems as a point set can be ordered with respect to some line (such that each point of the point set has a unique projection on the line) yielding some sort of "wiggled" spine; see Fig. 2.6b. In particular, point sets for the problems of drawing edges with circular arcs [15] and polylines with at most one bend [89, 130] have been defined based on the existence of monotone and down-up monotone arc diagrams, respectively; see also Fig. 2.6c. Finally, monotone arc diagrams have also been considered for directed graphs in the literature as *upward topological book embeddings* [67] in their own interest.

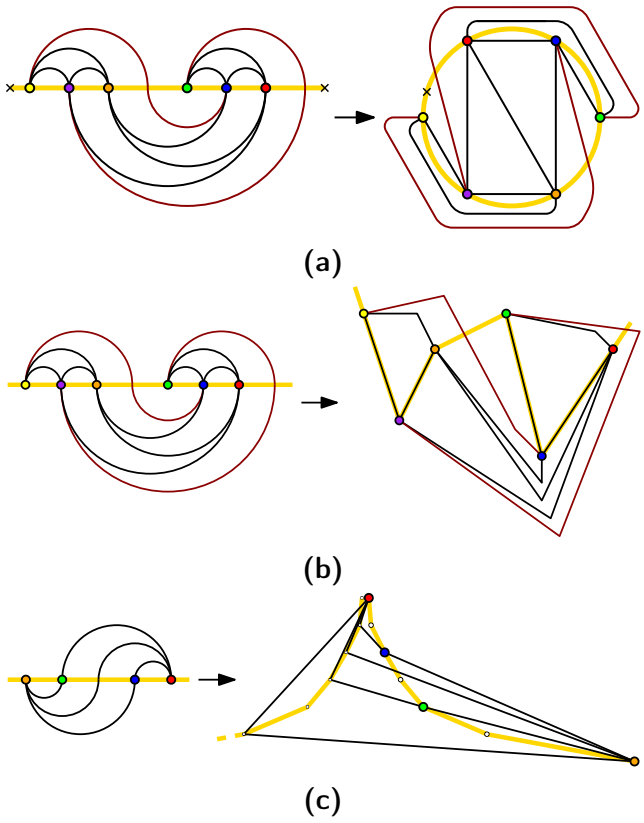


Figure 2.6: Three applications of arc diagrams: (a) By gluing together both ends of the spine, a circular layout is obtained in which biarcs (red) cross the circle on which the vertices are located. (b) By ordering the points of a general point set with respect to some orientation and using this ordering as a "wiggled" spine, a drawing with at most two bends per edge (occuring on the red biarcs) can be achieved [120]. (c) Conversion from a down-up monotone arc diagram to a drawing with one bend per edge on an universal point set as described in [130]. Note that the positions of points in the figure diverge from the exact positions according to [130] for easier readability.

Part I

Beyond Orthogonal Drawings

Smooth Orthogonal Drawings, Octilinear Drawings and Beyond Planar Graphs

Chapter 3

Smooth Orthogonal and Octilinear Drawings of Planar Graphs

In this chapter[1], we consider two extensions of the well-established orthogonal graph drawing model: (i) the *smooth orthogonal* graph drawing style, in which edges are represented by sequences of circular arcs and axis-aligned straight-line segments where consecutive segments have a common tangent at their touching point, and, (ii) the *octilinear* graph drawing style, where in addition to axis aligned straight-line segments also diagonals at slopes ± 1 are allowed. In particular, we focus on such drawings that have a low *curve complexity*, that is, the representations of edges are composed of few arcs or segments. In addition, in this chapter, we will only consider planar drawings, hence, we will omit to specifically mention that a drawing is planar. First we show in Section 3.1, that the classes of graphs that admit SC_1 and $8C_1$ drawings, respectively, are incomparable. Then, in Section 3.2, we show, that the topology-shape-metrics framework which is of fundamental importance in orthogonal graph drawing cannot be extended to either drawing style since in both cases no efficient algorithm for the metrics step can exist unless $P = NP$. Finally, in Section 3.3, we present efficient drawing algorithms for creating smooth orthogonal and octilinear Kandinsky layouts of planar graphs of arbitrary maximum degree.

3.1 Relations

In this section, we investigate the relationships between the classes of graphs that admit smooth orthogonal and octilinear drawings of low curve complexity. The primal motivation for investigating these relationships is the observation that

[1] The results of this chapter also appeared in [26].

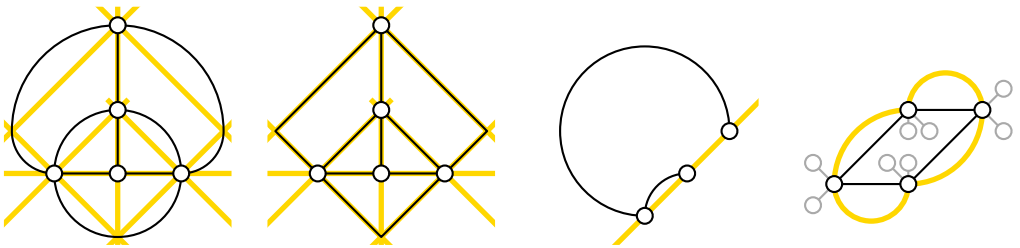


Figure 3.1: (a)–(b) A smooth orthogonal and an octilinear drawing of the same graph, respectively, in which vertices and bends have the same position. (c)–(d) A smooth orthogonal and an octilinear drawing of graph, respectively, whose underlying graph does not admit an octilinear or smooth orthogonal drawing with the same vertex position, respectively.

smooth orthogonal layouts and octilinear layouts require the same relative positions for the endpoints of edges. More precisely, if two points p and q (endpoint or bend) are connected by a segment of an edge, p and q have to be located on a horizontal or vertical line (for horizontal and vertical segments in both models as well as half circular arcs in the smooth orthogonal model) or on a line of slope ± 1 (for diagonal segments in the octilinear model as well as quarter and three-quarters circular arcs in the smooth orthogonal model). This similar positioning of vertices may be used to convert a drawing of one type into a drawing of the other type by replacing circular arcs with diagonal segments, or vice versa; for an example refer to the two drawings in Figs. 3.1a and 3.1b.

However, such a trivial conversion is not always possible: For instance, consider the smooth orthogonal drawing in Fig. 3.1c. If we were to replace both edges represented by circular arcs we would obtain a drawing with two overlapping edges. Conversely, the octilinear drawing in Fig. 3.1d cannot be converted with fixed vertex positions as well since the 4-cycle illustrated with black edges cannot be realized in the smooth orthogonal model such that the correct number of ports is free to the inner and outer face at each vertex. On the other hand, it is easy to see that the graphs shown in Figs. 3.1c and 3.1d indeed admit both a smooth orthogonal and an octilinear drawing. As a consequence, we may ask whether a less trivial conversion from one drawing style to the other may still be possible.

For the sake of brevity, let SC_k and $8C_k$ denote the class of graphs that admit an SC_k or $8C_k$-drawing, respectively. By definition, it holds that $SC_1 \subseteq SC_2$ and $8C_1 \subseteq 8C_2 \subseteq 8C_3$. Moreover, it is known that all planar graphs of maximum degree eight admit an $8C_3$-drawing [121] whereas all planar graphs of maximum degree four admit both an SC_2- [9] and an $8C_2$-drawing [28]. In addition, it was shown in [28] that there exist graphs of maximum degree 6 that admit an $8C_3$-drawing but no $8C_2$-drawing. Since there exists planar graphs of maximum degree five that admit $8C_2$-drawings but no $8C_1$-drawings [34], it holds that $8C_2$ is a superclass of

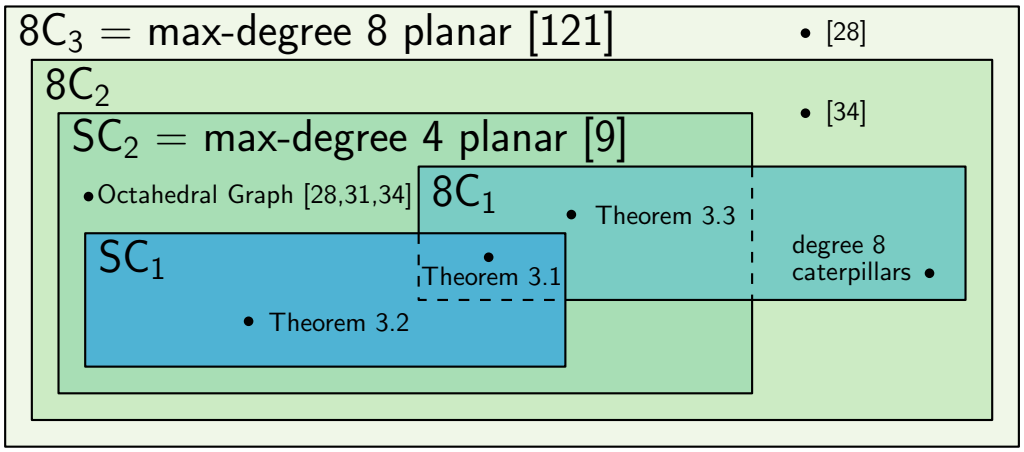


Figure 3.2: Overview of the relationships between the classes of graphs admitting SC_1-, SC_2-, $8C_1$-, $8C_2$- and $8C_3$-drawings.

both SC_2 and $8C_1$. The octahedral graph has been shown to admit neither an SC_1- [31] nor an $8C_1$-drawing [34], however, as a planar graph of maximum degree four, it admits both an SC_2- and an $8C_2$-drawing. Finally, it is easy to see that a caterpillar whose spine vertices have degree eight admits an $8C_1$-drawing whereas its degree is too large to admit an SC_2-drawing.

In the remainder of the section, we will first show in Theorem 3.1, that there are infinitely many planar 4-regular graphs that admit both an SC_1-drawing and an $8C_1$-drawing. In fact, these graphs will admit drawings in the two drawing styles that can be "converted" into each other as discussed before by the replacement of circular arcs with straight-line segments, and vice versa. On the other hand, we will show in Theorems 3.2 and 3.3, that there are graphs belonging to classes SC_1 and $8C_1$ that do not belong to the other class. As a result, our analysis will establish the relationships between graph classes SC_1, SC_2, $8C_1$, $8C_2$ and $8C_3$ as depicted in Fig. 3.2.

Theorem 3.1. *For every $k \in \mathbb{N}_+$, there exists a 4-regular planar graph G_k on $20k$ vertices that admits both an SC_1- and an $8C_1$-drawing.*

Proof. Refer to Fig. 3.3 for an illustration for the case where $k = 2$. We describe G_k for $k \in \mathbb{N}_+$: For $1 \leq i \leq 2k$ and $j \in \{t, b\}$, graph G_k contains a subgraph $W_{i,j}$ which is a 5-wheel with center $c_{i,j}$ and rim $(n_{i,j}, w_{i,j}, s_{i,j}, e_{i,j})$. Note that in Fig. 3.3, we layout $c_{i,j}$ as the central vertex of $W_{i,j}$, $n_{i,j}$ as the topmost vertex of $W_{i,j}$, $w_{i,j}$ as the leftmost vertex of $W_{i,j}$, $s_{i,j}$ as the bottommost vertex of $W_{i,j}$ and $e_{i,j}$ as the rightmost vertex of $W_{i,j}$. In addition, graph G_k contains the following edges:

- For $1 \leq i \leq 2k - 1$ and $j \in \{t, b\}$, we have that $(e_{i,j}, w_{i+1,j})$ is an edge of G_k; see the blue dotted edges in Fig. 3.3.

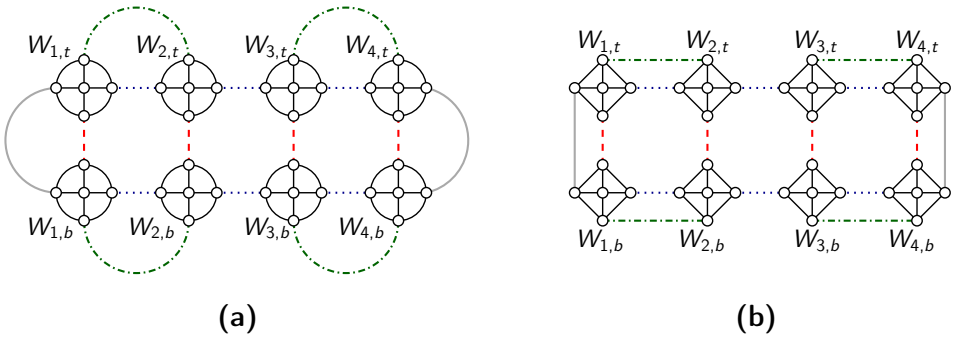


Figure 3.3: (a) SC_1- and (b) $8C_1$-drawing of G_2 as described in the proof of Theorem 3.1.

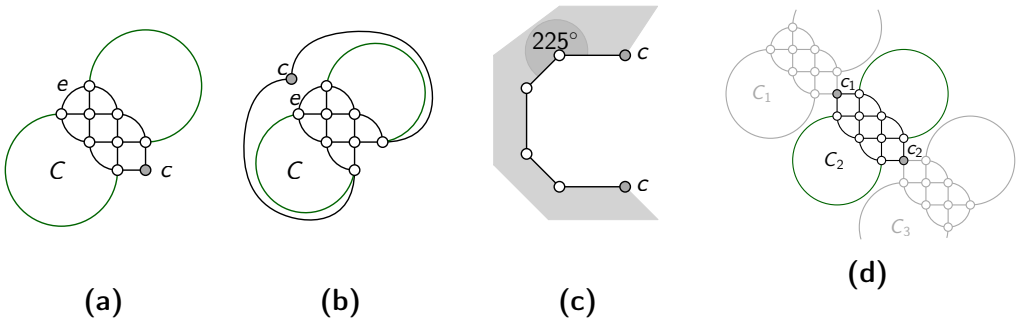


Figure 3.4: (a) SC_1-drawing of subgraph C, (b) alternative embedding of C, (c) illustration for the proof that C admits no $8C_1$-drawing with vertex c on the outer face, and, (d) construction for infinitely many 4-regular graphs containing two copies of C.

- For $0 \leq i \leq k-1$, we have that $(n_{2i+1,t}, n_{2i+2,t})$ and $(s_{2i+1,b}, s_{2i+2,b})$ are edges of G_k; see the green dash-dotted edges in Fig. 3.3.
- For $1 \leq i \leq 2k$, we have that $(s_{i,t}, n_{i,b})$ is an edge of G_k; see the red dashed edges in Fig. 3.3.
- $(w_{1,t}, w_{1,b})$ and $(e_{2k,t}, e_{2k,b})$ are edges of G_k; see the gray solid edges in Fig. 3.3.

It is straight-forward to verify that G_k is 4-regular. In addition, Figs. 3.3a and 3.3b certify that G_k admits both an SC_1- and an $8C_1$-drawing, respectively. □

Theorem 3.2. *For every $k \in \mathbb{N}_0$, there exists a 4-regular planar graph G_k on $9k+17$ vertices that admits an SC_1-drawing but no $8C_1$-drawing.*

Proof. Consider the graph C shown in Fig. 3.4a. Clearly, C admits a SC_1-drawing where vertex c (gray in Fig. 3.4a) is on the outer face such that c also has two free ports on the outer face. We observe that the graph obtained by removing c

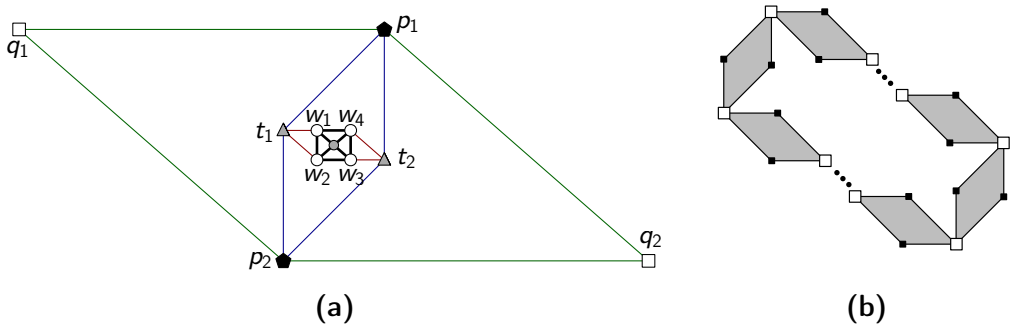


Figure 3.5: (a) Subgraph B that forms the basic component of the proof of Theorem 3.3, and, (b) a cycle of copies of B.

and connecting its two neighbors by an edge is triconnected, hence C has a unique embedding up to the choice of the outer face. In particular, the two embeddings shown in Figs. 3.4a and 3.4b are the only two embeddings where c is on the outer face.

In the following, we show that there is no $8C_1$-drawing of C where c is located on the outer face. Since the only two possible embeddings are those in Figs. 3.4a and 3.4b, the outer face has length 4 or 5 (Property 1). In addition, we observe, that in both embeddings, every vertex of the outer face except for c has two edges that use ports pointing in the interior of C. As a result, the angle formed by two consecutive edges on the outer face except for the pair of edges incident to c is at most 225° (Property 2). It is easy to see, that it is not possible to draw the outer face without any bends while satisfying Properties 1 and 2; see Fig. 3.4c. As a result, we conclude that C does not admit an $8C_1$-drawing with c on the outer face.

Finally, based on graph C, we construct a 4-regular planar graph G_k for $k \in \mathbb{N}_0$ that consists of $k+2$ biconnected components $C_1, \ldots, C_{k+2}$ arranged in a chain; see Fig. 3.4d for an illustration of G_1. In particular, C_1 and C_{k+2} are isomorphic to C while components $C_2, \ldots, C_{k+1}$ are isomorphic to a graph C' where edge e of graph C is split by a vertex c_1. Since there are two copies of C in each graph, one of them has to be drawn with c on the outer face which is not possible in an $8C_1$-drawing. On the other hand, Fig. 3.4d certifies that an SC_1-drawing exists, hence, we conclude the proof. □

Theorem 3.3. *For every $k \in \mathbb{N}_0$, there exists a 4-regular planar graph G_k on $20k + 40$ vertices that admits an $8C_1$-drawing but no SC_1-drawing.*

Proof. The central ingredient of our proof is the graph B shown in Fig. 3.5a. We first describe the structure of B: First, B contains a 5-wheel W_5 composed of center c (gray circle in Fig. 3.5a) and rim $(w_1, \ldots, w_4)$ (white circles in Fig. 3.5a). Attached to W_5, there are two triangular faces (t_1, w_1, w_2) and (t_2, w_3, w_4) (t_1 and t_2 are depicted as triangles in Fig. 3.5a). Note that t_1 and t_2 form a separation pair.

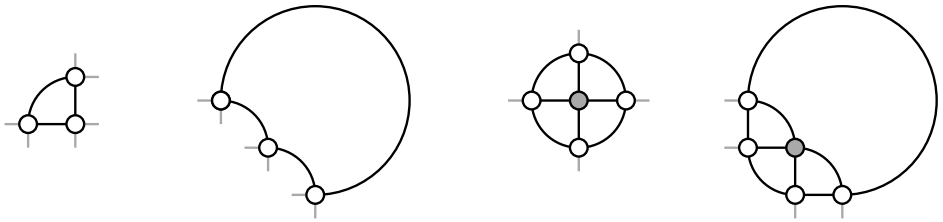


Figure 3.6: (a)–(b) All possible embeddings of a triangle with all free ports on the outer face, and, (c)–(d) all possible embeddings of W_5 with all free ports on the outer face.

Moreover, both t_1 and t_2 are connected to vertices p_1 and p_2 (depicted as pentagons in Fig. 3.5a) creating pentagons $(p_1, t_1, w_1, w_4, t_2)$ and $(p_2, t_2, w_3, w_2, t_1)$. Again, p_1 and p_2 form a separation pair and are both incident to vertices q_1 and q_2 (depicted as quadrilaterals in Fig. 3.5a) yielding two quadrilateral faces (q_1, p_2, t_1, p_1) and (q_2, p_1, t_2, p_2). As a result, B has two vertices of degree 2 (that is, q_1 and q_2, and two separation pairs, that is, (t_1, t_2) and (p_1, p_2). All remaining vertices have degree exactly 4. It is noteworthy that when the outer face is required to be (q_1, p_1, q_2, p_2) (which we will require later), the separation pairs only allow flips of some symmetric triconnected components which just results in a renaming of vertices.

For $k \in \mathbb{N}_0$, we construct a 4-regular planar graph G_k that consists of a cycle of $2k + 4$ copies of B where consecutive copies B and B' are attached to each other by identifying vertex q_2 of B with vertex q_1 of B'; see Fig. 3.5b in which copies of B are outlined as gray-shaded parallelograms. As certified by Fig. 3.5, G_k admits an $8C_1$-drawing. Moreover, by planarity, all but one copy of B must be realized such that (i) (q_1, p_1, q_2, p_2) is the outer face, and (ii) q_1 and q_2 have two free ports on the outer face (to identify them with the corresponding vertex of another copy). In the following, we show that these two properties cannot be achieved in any SC_1-drawing of B which directly implies that for any k, G_k has no SC_1-drawing. Note that since we require (q_1, p_1, q_2, p_2) to be the outer face, the embedding of B must be isomorphic to the embedding shown in Fig. 3.5a.

We start by investigating wheel W_5. Since the embedding of W_5 is fixed, all unoccupied ports have to be located on the outer face; the same is true for the four triangles W_5 is composed of. As shown in [9], there are only two SC_1-drawings of a triangle fulfilling this property; see Figs. 3.6a and 3.6b. It is easy to see that these two realizations can only be combined into two different drawings of W_5 (up to isomorphy) which are shown in Figs. 3.6c and 3.6d.

Next, consider vertices t_1 and t_2. Each of them creates a triangular face with two vertices of W_5 which again has to have all free ports on the outer face. As a result, we have to draw each such face with one of the drawings shown in Figs. 3.6a

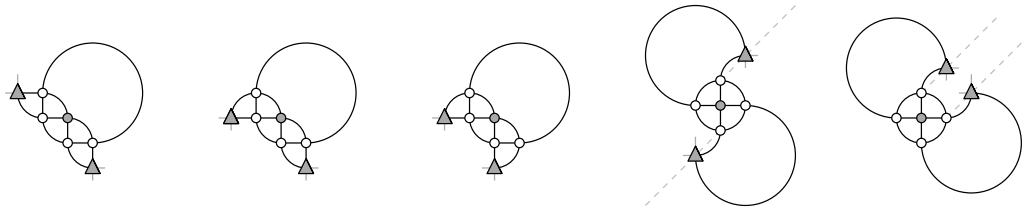


Figure 3.7: All SC_1-drawings of the subgraph of B induced by the vertices of W_5, t_1 and t_2 that might be extendable to a drawing of B.

and 3.6b. It is straight-forward to see that there are only five different drawings of the subgraph induced by the vertices of W_5, t_1 and t_2 which are shown in Fig. 3.7. Note that the geometry of the drawings shown in Figs. 3.7d and 3.7e can be altered by moving vertices t_1 and t_2 along the gray dashed diagonal rays.

In the next step, we insert p_1 and p_2 into the drawings of W_5, t_1 and t_2. We do so by considering all candidate positions, that we identify as follows: In an SC_1-drawing, both endpoints of an edge are located on a common horizontal, vertical or diagonal (at slope ± 1). Since both t_1 and t_2 are neighbors of p_1 and p_2 this limits the number of feasible candidate positions as we only have to consider all intersections of pairs of rays of slopes $\{0, 1, -1, \infty\}$ which emerge from t_1 and t_2, respectively. This allows us to enumerate all possible candidate positions for p_1 and p_2; see the gray-colored pentagons in Figs. 3.8 and 3.9. It is noteworthy, that for the case where W_5, t_1 and t_2 use the drawing in Fig. 3.7e there are four different subcases to be considered. These subcases depend on the relative positioning of t_1 and t_2 that we assume to be fixed in the following; see Fig. 3.9a. Observe that there are symmetric relative positionings of t_1 and t_2 with respect to the diagonal line through c (dashed-dotted in Fig. 3.9a) indicated by the identical numbering in Fig. 3.9a. In Figs. 3.9b to 3.9e we then illustrate the non-symmetrical ones that are marked with an asterisk in Fig. 3.9a. In particular, in Fig. 3.9b, t_1 and t_2 are diagonally aligned, in Fig. 3.9d, they are vertically aligned (which is symmetrical to the case where they are horizontally aligned). Figs. 3.9c and 3.9e show the two remaining cases, that is, the case where t_2 is located between a vertical and a diagonal ray through t_1, and the case where t_2 is located to the right of the vertical line through t_1, respectively.

For each candidate position, we exhaustively try all possible realizations for edges from t_1 and t_2 to p_1 (or p_2, respectively) using a single edge segment allowed by the smooth orthogonal model. We then classify a candidate position as *valid* if and only if none of the following *forbidden patterns* arises:

FP.I. one of the two edges cannot be drawn without intersections

FP.II. a port has to be used twice to realize the two edges

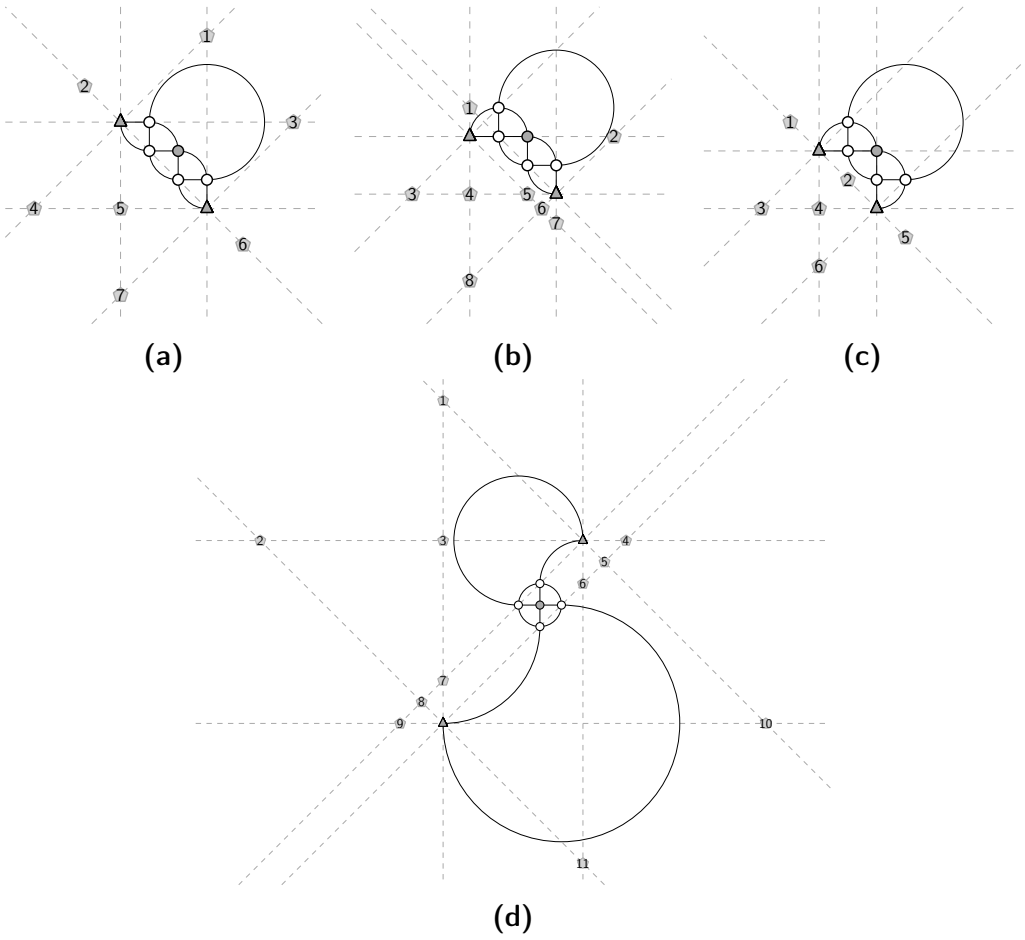


Figure 3.8: Candidate positions for p_1 and p_2 for the case where W_5, t_1 and t_2 are drawn as shown in (a) Fig. 3.7a, (b) Fig. 3.7b, (c) Fig. 3.7c, and, (d) Fig. 3.7d.

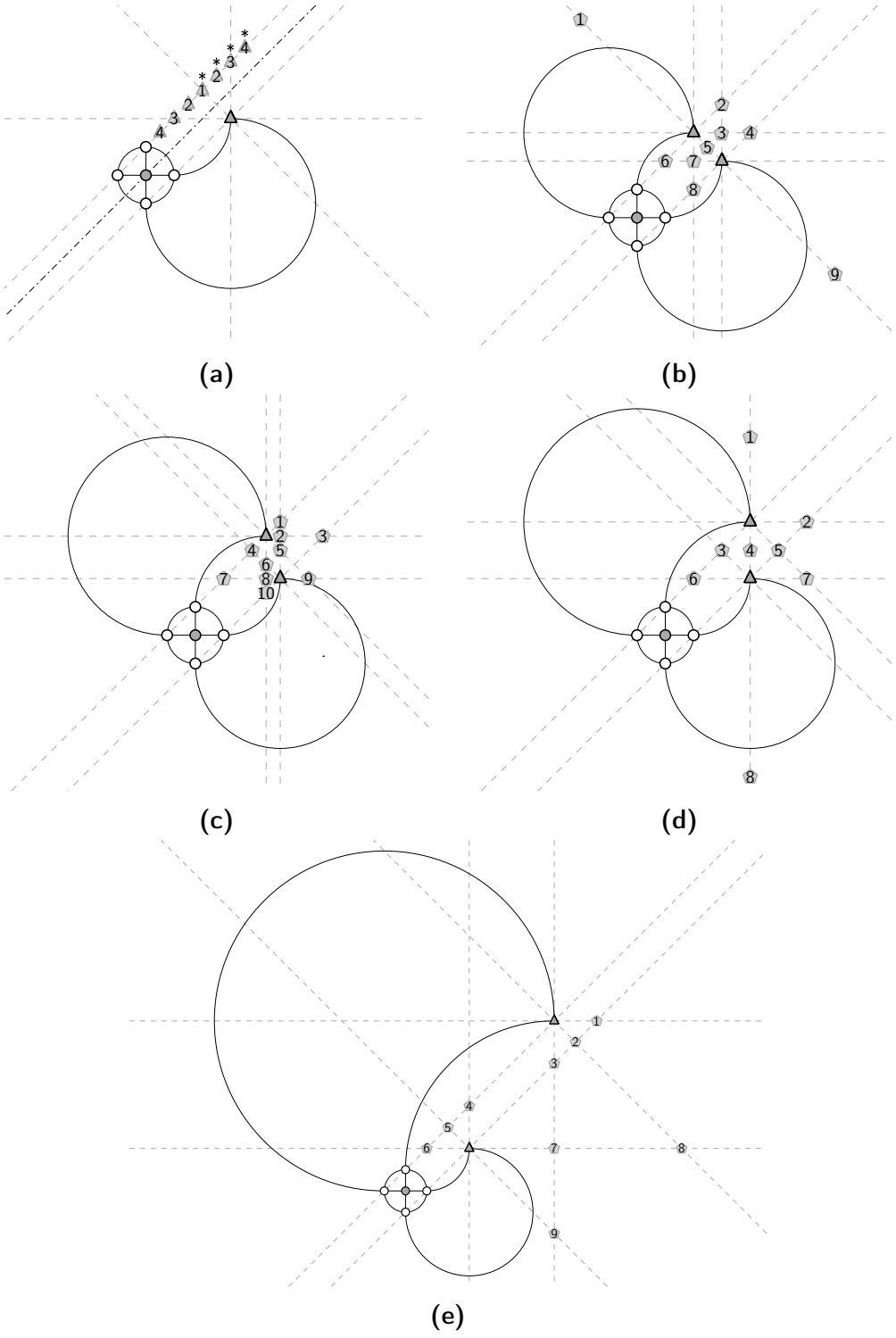


Figure 3.9: Candidate positions for p_1 and p_2 for the case where W_5, t_1 and t_2 are drawn as shown in Fig. 3.7e. Subfigure (a) shows possible relative positionings of t_1 and t_2 (non-symmetric ones marked with an asterisk). Subfigures (b)–(e) show candidate positions arising for all non-symmetric relative positionings of t_1 and t_2.

Table 3.1: Forbidden patterns arising when using each of the candidate positions shown in Figs. 3.8 and 3.9 for placing p_1 or p_2. The gray-shaded row lists forbidden patterns arising when placing q_1 and q_2 at each of the candidate positions derived from the only valid drawing of B without q_1 and q_2 which is illustrated in Fig. 3.10 and obtained by placing p_1 and p_2 at Positions 2 and 6 of Fig. 3.8a, respectively. Table entries of '–' indicate valid candidate positions causing no forbidden pattern.

Case	Candidate Position (CP.)/ Forbidden Pattern (FP.)										
Fig. 3.8a	CP.:	1	2	3	4	5	6	7			
	FP.:	III	–	III	III	III	–	III			
Fig. 3.8b	CP.:	1	2	3	4	5	6	7	8		
	FP.:	III	II	–	III	III	III	–	–		
Fig. 3.8c	CP.:	1	2	3	4	5	6				
	FP.:	III	III	III	–	III	III				
Fig. 3.8d	CP.:	1	2	3	4	5	6	7	8	9	10	11
	FP.:	III	II	II	III	III	I	I	III	III	II	III
Fig. 3.9b	CP.:	1	2	3	4	5	6	7	8	9		
	FP.:	I	II	III	II	–	II	III	II	I		
Fig. 3.9c	CP.:	1	2	3	4	5	6	7	8			
	FP.:	I	II	III	–	III	II	II	I			
Fig. 3.9d	CP.:	1	2	3	4	5	6	7	8	9	10	
	FP.:	II	III	II	III	III	III	II	III	II	II	
Fig. 3.9e	CP.:	1	2	3	4	5	6	7	8	9		
	FP.:	II	III	III	III	III	II	III	II	II		
Fig. 3.10	CP.:	1	2	3	4	5	6	7	8	9		
	FP.:	I	III	III	I	I	III	I	III	I		

FP.III. a vertex has an unoccupied port incident to an interior face (this will prevent q_1 and q_2 from appearing on the outer face)

If a candidate position is not valid, we call it *invalid*. We point out that we chose the radii of arcs incident to t_1 and t_2 in Figs. 3.8d and 3.9 in a way that avoids the creation of unnecessary Forbidden Patterns I.

In Table 3.1, we list for each candidate position for p_1 and p_2 as shown in Figs. 3.8 and 3.9 which forbidden pattern arises (if any). It is immediate to see that all drawings for W_5, t_1 and t_2 shown in Figs. 3.8c–3.8d as well as in Figs. 3.9b–3.9e provide at most one valid candidate position for p_1 and p_2. As a result, those drawings cannot be subdrawings of any SC_1-drawing of B.

The drawing shown in Fig. 3.8b, on the other hand, provides three valid can-

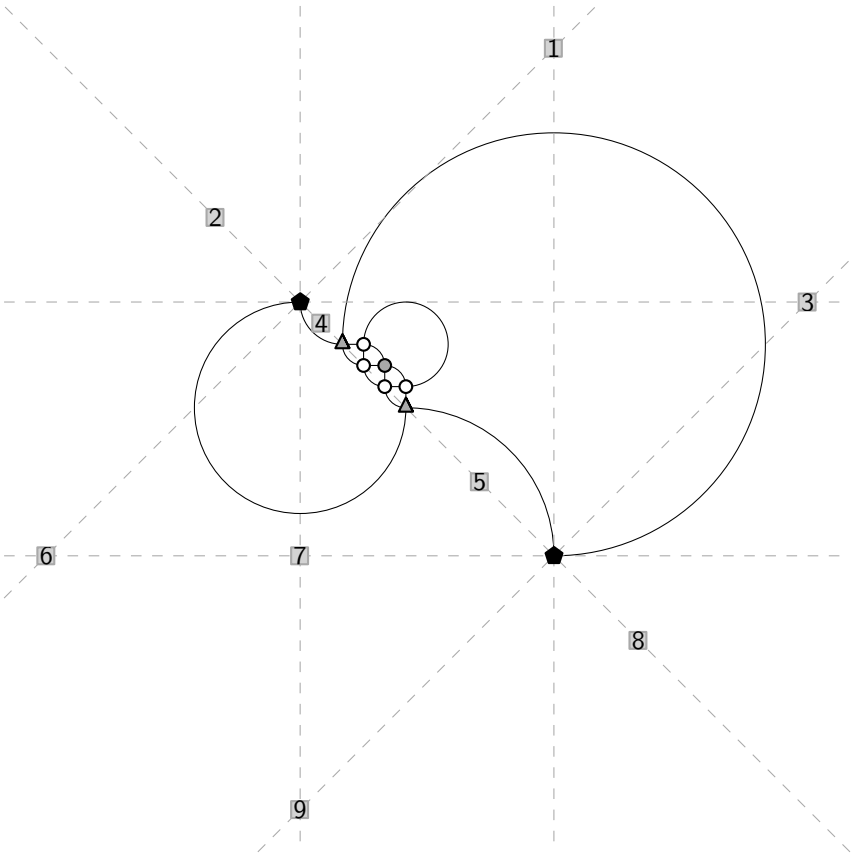


Figure 3.10: Valid SC_1-drawing of B without q_1 and q_2 and candidate positions for placement of q_1 and q_2.

didate positions, namely Positions 3,7 and 8. This gives rise to three different combinations of placements for vertices p_1 and p_2, however each of them will result in a Forbidden Pattern II. It remains the option to place vertices p_1 and p_2 on Candidate Positions 2 and 6 in Fig. 3.8a, respectively. This indeed results in a valid drawing as shown in Fig. 3.10.

Finally, as we did before for p_1 and p_2, we also consider all possible candidate positions for placing q_1 and q_2; see the gray squares in Fig. 3.10. Again, when choosing the radii for arcs incident to p_1 and p_2 in Fig. 3.10, we ensure that no unnecessary forbidden configuration arises. Nevertheless, it turns out, that all candidate positions for q_1 and q_2 are invalid; see the gray shaded row in Table 3.1 for the corresponding forbidden patterns.

As a result, there exists no SC_1-drawing of B where both q_1 and q_2 are on the outer face having two free ports. This concludes the proof of the theorem. □

3.2 *NP*-Hardness of the Metrics Step

In this section, we investigate the complexity of finding SC_1- and $8C_1$-drawings for planar graphs of maximum degree four. Recall that planar graphs of maximum degree three always admit both an SC_1- and an $8C_1$-drawing [29, 69, 116] while it is known that finding an $8C_1$-drawing for planar graphs of maximum degree eight is NP-hard [135]. Here, we make an additional assumption on the input of our drawing algorithm: We assume, that in addition to an input graph G, we are also provided with an embedding of G and a *smooth orthogonal* or an *octilinear representation* $\mathcal{R}$. Such a representation is defined analogously to an orthogonal representation, that is, the input of the last step (that is, the Metrics step) of the topology-shape-metrics framework for orthogonal graph drawing. More precisely, it defines

(i) the angles between edges appearing consecutively around a vertex in the cyclic order (which are multiples of $\pi/2$ for the smooth orthogonal and multiples of $\pi/4$ for the octilinear setting), and,

(ii) the *shape* of each edge, that is whether it is drawn as a straight-line segment (always in the octilinear setting) or as a circular arc (only in the smooth orthogonal setting) in which case it is specified, if the circular arc is a quarter, half, or three-quarters circular arc and with which sign the slope of its tangent changes when exiting either endpoint.

In contrast to the orthogonal graph drawing model, the two theorems in this section show that in both the smooth orthogonal and the octilinear model, the metrics step can only be efficiently solved if $P = NP$. In addition, we remark that the problems studied in this section are also closely related to *HV-rectilinear planarity testing* [131]. More precisely, in HV-rectilinear planarity testing, the input assigns a label to each edge of an input graph that describes whether it should be drawn as a horizontal or as a vertical segment. In contrast to our two problems, HV-rectilinear planarity testing is polynomial-time solvable if the input graph has a fixed embedding [83] and only becomes NP-hard in the variable embedding setting [73].

Theorem 3.4. *Given a planar graph G of maximum degree four with a corresponding smooth orthogonal representation $\mathcal{R}$, it is NP-hard to decide whether G admits a smooth orthogonal drawing realizing $\mathcal{R}$. This holds even if $\mathcal{R}$ specifies that all edges are to be drawn as straight-line segments or quarter circular arcs.*

Proof. Our reduction is from the well-known NP-hard problem 3-SAT [99]: The input of 3-SAT is a boolean formula φ in conjunctive normal form, that is, it is a conjunction ($\wedge$) of disjunctive clauses ($\vee$). In particular, in 3-SAT each clause is the disjunction of exactly three literals (which are either a variable x or a negated variable $\neg x$). More formally, φ consists of a set of variables X and a set of clauses

C where every $c \in C$ is a set of three literals $l_1(c), l_2(c), l_3(c)$ taken from the literal set $L = X \dot\cup \{\neg x | x \in X\}$. The problem then asks whether there is an assignment of truth values $t : X \to \{\bot, \top\}$ such that for every $c \in C$ it holds that $t(l_1(c)) \vee t(l_2(c)) \vee t(l_3(c)) = \top$ where for every $l \in L$, $t(l) = t(x)$ if $l = x$ for $x \in X$ or $t(l) = \neg t(x)$ if $l = \neg x$ for $x \in X$. If that is the case, we call φ *satisfiable*, otherwise we call it *unsatisfiable*.

For our reduction, we assume to be given a 3-SAT formula φ that we encode in a graph G_φ with a smooth orthogonal representation $\mathcal{R}_\varphi$ in such a way that G_φ admits an SC_1-drawing Γ_φ if and only if φ is satisfiable; for an example refer to Fig. 3.11.

The following are the main ideas of our reduction:

(i) We encode *information* about the truth values of literals in the length $\ell(e)$ of certain straight-line edges e of Γ_φ.

(ii) We use rectangular faces of Γ_φ to propagate the information stored in the length of one of its sides to the opposite side of the rectangular face, that is, the information *flows* across rectangular faces.

(iii) We use triangular faces composed of two straight-line edges and a quarter circular arc to change the *direction* of the *information flow* as those faces have the property that both straight-line edges have the same length while they are perpendicular.

(iv) We propagate a unit length $\ell(u)$ that can be used as an input for all of our gadgets. This allows us to communicate the meaning of edge lengths (true literal versus false literal) amongst disjoint gadgets.

We will make use of those techniques in the construction of our gadgets that we describe next.

Variable gadgets. For each variable $x \in X$, we introduce a *variable gadget* as illustrated in Fig. 3.12. We ensure with the bold-drawn circular arc that the sum of edge lengths to its left is the same as the sum of the edge lengths to its bottom; see the gray vertices in Fig. 3.12. The input of the gadget are three unit length edges, which ensures that the sum of lengths of output edges x and $\neg x$ is equal to three times the unit edge length, i.e., $\ell(x) + \ell(\neg x) = 3\ell(u)$.

Assuming that the lengths of all straight-line edges were integer and at least one and further assuming that $\ell(u) = 1$, this construction would already suffice to ensure that $\ell(x), \ell(\neg x) \in \{1, 2\}$, i.e., there would be two disjoint states. Then, we could define that the assignment $\ell(x) = 2$ and $\ell(\neg x) = 1$ corresponds to assignment $x = \top$ while we could define the assignment $\ell(x) = 1$ and $\ell(\neg x) = 2$ to correspond to assignment $x = \bot$. However, if $\ell(u) = 2$, a drawing algorithm could chose $\ell(x) = \ell(\neg x) = 3$, indicating that both x and $\neg x$ are *"half true"*. In the

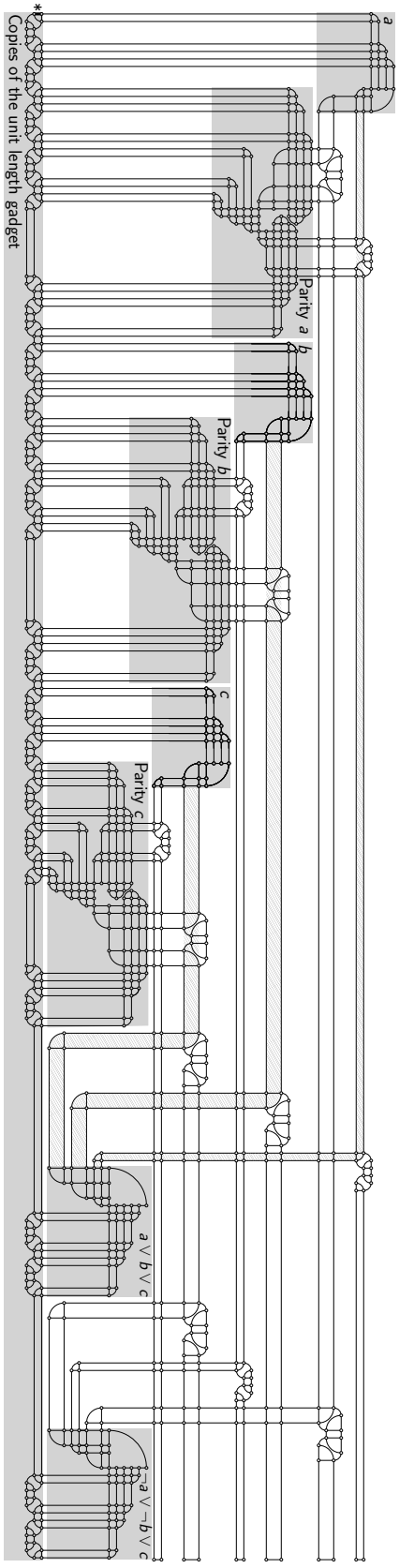


Figure 3.11: Drawing Γ_φ for φ consisting of variable set $X = \{a, b, c\}$ and clause set $C = \{\{a, b, c\}, \{\neg a, \neg b, c\}\}$. The drawing shows that there exists a satisfying truth value assignment t with $t(a) = \bot$ and $t(b) = t(c) = \top$.

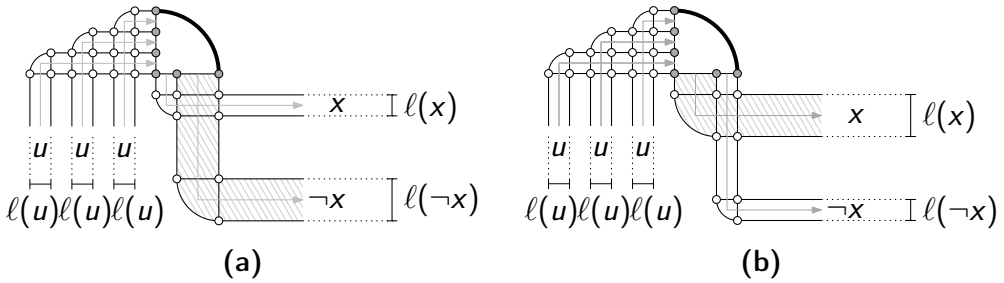


Figure 3.12: The variable gadget used by our reduction. Gray arrows show the information flow. (a) True state $x = \top$: $\ell(x) \approx 2\ell(u)$ and $\ell(\neg x) \approx 1\ell(u)$. (b) False state $x = \bot$: $\ell(x) \approx 1\ell(u)$ and $\ell(\neg x) \approx 2\ell(u)$.

following, we solve this issue by introducing a *parity gadget* for each variable, which allows us to use real valued edge lengths while also ensuring that $\ell(x), \ell(\neg x) \in \{\ell(u) + \varepsilon, 2\ell(u) - \varepsilon\}$ for some $\varepsilon << \ell(u)$.

Parity gadgets. In addition to variable gadgets, we introduce a *parity gadget* for each variable $x \in X$ that will introduce an intersection in Γ_φ, if the values of $\ell(x)$ and $\ell(\neg x)$ are not significantly different. We illustrate the gadget in Fig. 3.13 in which the central detail, referred to as *vertical gap* is shaded in gray. The vertical gap is a gap of width $3\ell(u)$ into which two blocks consisting of two square-shaped and three triangular faces point inside from either side; see triangular- and square-shaped vertices in Figs. 3.13a–3.13c or Fig. 3.13d for a more detailed illustration of the vertical gap. Depending on the choice of $\ell(x)$, which as discussed before fixes $\ell(\neg x) = 3\ell(u) - \ell(x)$, one of the blocks may be located above the other. In particular, if $\ell(x) \approx 2\ell(\neg x)$, the right block is above the left block; see Fig. 3.13a. In contrast, if $\ell(x) \approx \frac{1}{2}\ell(\neg x)$, the left block is above the right block; see Fig. 3.13a. On the other hand, if $\ell(x) \approx \ell(\neg x)$, both blocks overlap and intersections occur; see Fig. 3.13c. As a result, in any drawing Γ_φ realizing $\mathcal{R}_\varphi$, it holds that $\ell(x) \not\approx \ell(\neg x)$.

In the following, we give a more precise bound for the difference between $\ell(x)$ and $\ell(\neg x)$ that is needed in order to avoid intersections. We consider only the case where $x = \bot$ as the other case is symmetric. Here, we have to avoid that the two quarter circular arcs intersected by the dashed diagonal in Fig. 3.13d are involved in intersections, that is, the top circular arc has to be entirely located above the bottom one. By the design of the gadget, both circular arcs have radius $\ell(u)$, that is, their centers (colored in gray in Fig. 3.13d) must have an Euclidean distance of more than $2\ell(u)$ which is equal to the length of the diagonal dashed line-segment. Using the Pythagorean theorem, we can also express the same length as $\sqrt{4\lambda^2 + \ell(u)^2}$ where $\lambda = \ell(\neg x) - \ell(x)$. As a result, in order to avoid intersections, it follows that $\lambda > \frac{\sqrt{3}}{2}\ell(u) \approx 0.866\ell(u)$ and hence in order to avoid intersections $\ell(x), \ell(\neg x) \in (0, 1.067\ell(u)) \cup (1.933\ell(u), 3)$, that is $\varepsilon < 0.067\ell(u) << \ell(u)$.

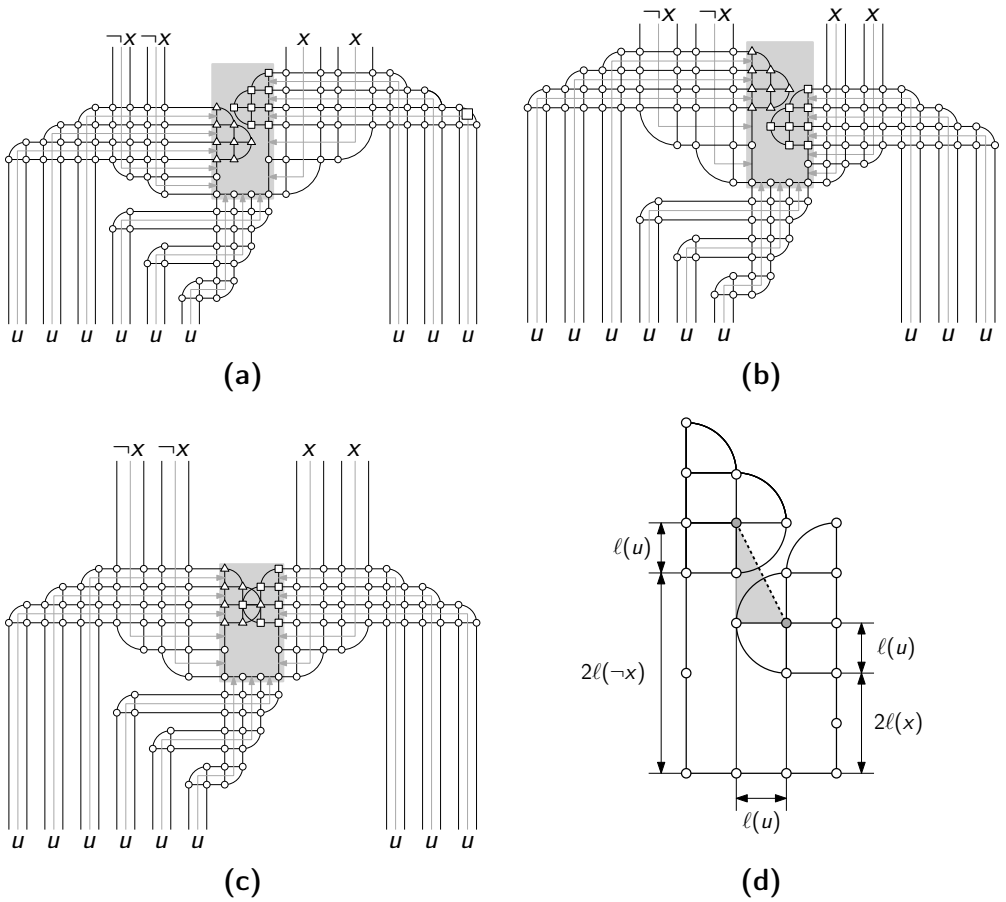


Figure 3.13: The parity gadget used by our reduction. Gray arrows show the information flow. (a) True state $x = \top$: $\ell(x) \approx 2\ell(u)$ and $\ell(\neg x) \approx 1\ell(u)$. (b) False state $x = \bot$: $\ell(x) \approx 1\ell(u)$ and $\ell(\neg x) \approx 2\ell(u)$. (c) Invalid state: $\ell(x), \ell(\neg x) \approx \frac{3}{2}\ell(u)$. (d) Central detail of the construction.

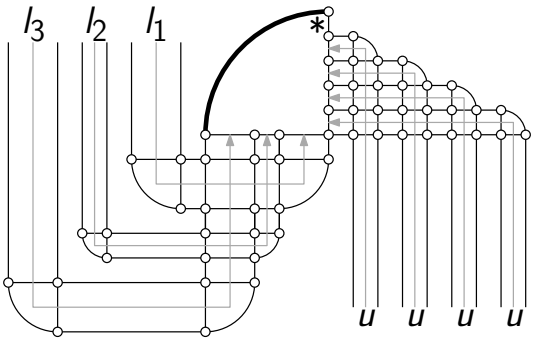


Figure 3.14: The clause gadget used by our reduction. Gray arrows show the information flow. The bold drawn arc can only be realized as a quarter circle if $\ell(l_1) + \ell(l_2) + \ell(l_3) > 4\ell(u)$.

Figure 3.15: Auxiliary gadgets used by our reduction. Gray arrows show the information flow. (a) The crossing gadget lets two flows of information cross. (b) The copy gadget creates three copies of an input information.

Clause gadget. Finally, for every clause $c \in C$ with literals l_1, l_2 and l_3, G_φ contains a *clause gadget* as shown in Fig. 3.14. As in the construction of the variable gadget, the bold-drawn quarter circular arc ensures equality between the two sums of information on the righthand side and on the bottom side. At the bottom-side, the three literals enter and hence the sum of edge lengths is equal to $\ell(l_1) + \ell(l_2) + \ell(l_3)$. On the righthand side on the other hand, we have four unit length edges as input in addition to a *free edge* (marked with an asterisk in Fig. 3.14). As a result, the sum of edge lengths at the righthand side is more than $4\ell(u)$ since the free edge has an arbitrary non-zero length.

We show that the quarter circular arc can be realized if and only if one of the three literals is true. Assume w.l.o.g. that for each variable, the length of the true literal is $2 - \varepsilon$ while the length of the false literal is $1 + \varepsilon$ for some ε chosen as large as possible (i.e., $\varepsilon < 0.067\ell(u)$). We will see that this assumption is indeed not a loss of generality as this allows use to choose the length of the unfulfilled literals as large as possible. Clearly, if at least one literal is true, it holds that $\ell(l_1) + \ell(l_2) + \ell(l_3) \geq 4\ell(u) + \varepsilon$ and the inequality holds. Then, the length of the free edge can be chosen such that the quarter circular arc can be realized. On the other hand, if no literal is fulfilled, $\ell(l_1) + \ell(l_2) + \ell(l_3) = 3\ell(u) + 3\varepsilon < 4\ell(u)$ and it is not possible to draw the bold drawn edge of Fig. 3.14 as a quarter circular arc. We conclude that the clause gadget can indeed verify whether a clause is satisfied or not.

Auxiliary Gadgets. In addition to the previously mentioned gadgets, our reduction uses two gadgets for routing the information flow within Γ_φ. First, we have the crossing gadget, which is simply a rectangular face which allows two flows of information i_1 and i_2 to cross; see Fig. 3.15a. In addition, we also have *copy gadgets* as depicted in Fig. 3.15b, that create three copies of an input information i. In particular, the copy gadget achieves that task by having two quadrilateral faces whose vertices are located at the vertices of a rectangle with sides of slopes ± 1; see the gray vertices in Fig. 3.15b. Finally, the unit length is provided by a *unit length gadget* which is a single edge which defines all unit lengths in our construction through a propagation via copy gadgets; see the edge marked with

an asterisk in Fig. 3.11.

Overall construction of G_φ and $\mathcal{R}_\varphi$. It remains to describe the entire construction; see Fig. 3.11 for an example drawing. We point out that the entire construction is rigid enough so that it can only be entirely rotated, as a result, we will describe the relative positioning of gadgets with terms such as "to the left" even though this can be altered by a rotation of the resulting drawing. Graph G_φ contains one unit length gadget that we copy $\mathcal{O}(|X| + |C|)$ times using copy gadgets. All these copy gadgets are located below the remainder of the construction in $\mathcal{R}_\varphi$. Then, for each $x \in X$, we create both a variable and a parity gadget which we connect to different copies of the unit length gadget. Moreover, we place the variable gadget to the top left of the corresponding parity gadget and connect the output literals of the variable gadget to the parity gadget with one copy gadget each. We enumerate all variables arbitrarily and position the variable and parity gadget of the i-th variable to the bottom right of the corresponding gadgets of the $(i-1)$-th variable. In addition, for each $c \in C$, we introduce a clause gadget that has four connections to different unit length copies. We position all clause gadgets in a row below the bottommost variable gadget. As a result we can connect the output literals of variable gadgets to the corresponding clause gadgets so that all crossings between literal informations appear above the clause gadgets. In particular, if a clause contains the i-th variable, there will be a crossing with all literals of variables with indices larger than i. All those crossings will be resolved with crossing gadgets. In total, for each clause, we add $\mathcal{O}(|X|)$ crossing and three copy gadgets. Since we can order variables and clauses arbitrarily in advance, the position of all required copy and crossing gadgets is fixed which allows us to compute G_φ and $\mathcal{R}_\varphi$ in $\mathcal{O}(|X||C|)$ time.

We complete the proof by showing that indeed there is a SC_1-drawing Γ_φ realizing $\mathcal{R}_\varphi$ if and only if ϕ is satisfiable. Assume that a valid drawing Γ_φ exists. Then, we can compute a satisfying truth value assignment t as follows: For each $x \in X$, we set $t(x) = \top$ if and only if $\ell(x) \geq 1.933\ell(u)$. Note that $\ell(u)$ can be easily determined by measuring the length of the unit length gadget. As we have shown before, for clause $c \in C$ with literals l_1, l_2, l_3, a clause gadget ensures that $\ell(l_1) + \ell(l_2) + \ell(l_3) > 4\ell(u)$, which can only be the case, if for one of the literals l_i it holds that $t(l_i) = \top$. As a result, t is satisfying φ. For the other direction of the proof, assume that there is a satisfying truth assignment t for φ. Here, we set $\ell(u) = 1$ and $\ell(x) = 1.95$ if $t(x) = \top$, or $\ell(x) = 1.05$ otherwise. As a result, for $c \in C$ with literals l_1, l_2, l_3 out of which at least one is true, it holds that $\ell(l_1) + \ell(l_2) + \ell(l_3) \geq 4.05 > 4\ell(u)$. This completes the proof. □

Next, we give an equivalent statement for the octilinear drawing problem.

Theorem 3.5. *Given a planar graph G of maximum degree four with a corresponding octilinear representation $\mathcal{R}$, it is NP-hard to decide whether G admits an*

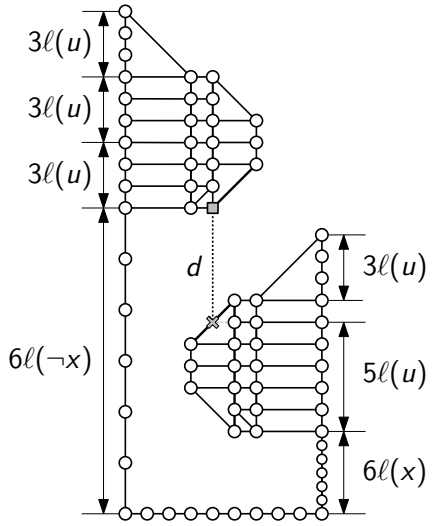


Figure 3.16: The vertical gap of the parity gadget used in the proof of Theorem 3.5. The illustration shows the case where $x = \bot$, i.e., $\ell(\neg x) \approx 2\ell(u)$ and $\ell(x) \approx \ell(u)$. The dotted line describes the the smallest vertical distance d between the top and bottom block.

octilinear drawing realizing $\mathcal{R}$. This holds even if $\mathcal{R}$ specifies that all edges are to be drawn without bends.

Proof. We follow the same proof outline as in the proof of Theorem 3.4. The adjustment to the octilinear model is achieved by replacing every quarter circular arc with a diagonal segment that uses one of the new ports in the octilinear model. This also maintains planarity by construction. The only gadget that needs more adjustments is the parity gadget; see Fig. 3.16. The routing of input information towards the vertical gap is done analogously, however the blocks are layed out differently.

Assume that $x = \bot$. It is straight-forward to see that the smallest vertical distance d between the top and bottom block in the vertical gap is equal to $6\ell(\neg x) - 6\ell(x) - 5\ell(u)$; see the dotted line-segment in Fig. 3.16. This implies that $\ell(\neg x) - \ell(x) > \frac{5}{6}\ell(u)$ as d must be positive. Hence, in the octilinear setting, $\varepsilon < 0.084\ell(u) << \ell(u)$. □

3.3 Bi-Monotone Kandinsky Drawings

In this section, we introduce new efficient algorithms for producing Kandinsky drawings in the smooth orthogonal and the octilinear model. In the smooth orthogonal model, only one drawing algorithm has been proposed so far [31]; the produced SC_2-Kandinsky drawings are obtained from planar arc diagrams and restrict vertices to one line and contain rather complicated edge representations consisting

of up to two half circular arcs. Here, we provide new algorithms that construct SC_2-drawings, which improves upon the previously known algorithm achieving the following aesthetic benefits:

(i) vertices are not restricted to a line but more equally spaced in the entire drawing area

(ii) each edge is *bi-monotone*, that is, both x- and y-monotone.

Our two algorithms will allow to opt either for quadratic area but an unbounded number of edges consisting of two segments or for cubic area and a spanning tree realized with edges consisting of one segment. We point out that not all planar graphs admit an SC_1-drawing in the Kandinsky model [29], hence, we achieve optimal curve complexity.

Theorem 3.6. *Let G be an n-vertex maximal planar graph. A planar bi-monotone SC_2-Kandinsky drawing of G in $\mathcal{O}(n^2)$ area can be computed in $\mathcal{O}(n)$ time.*

Proof. We use a modified version of the shift-method [60]. Let $\pi = (v_1, \ldots, v_n)$ be a canonical ordering of G. Our drawing algorithm will satisfy the following invariants for the drawing Γ_k of graph G_k:

I.1 All edges on C_k are quarter circular arcs except for edge (v_1, v_2) which is a horizontal line-segment; see Fig. 3.17a. As a consequence, both endpoints of an edge on C_k (except for (v_1, v_2)) are located on a line of slope ± 1.

I.2 Γ_k is planar.

I.3 Each vertex v on C_k is associated with a shift-set $S(v)$ such that edges between two shift-sets are blue or green (according to the Schnyder coloring obtained from the canonical ordering).

I.4 All blue and green interior edges in G_{k+1} are a sequence of a quarter circular arc and a horizontal segment (of possibly zero length).

We draw G_3 such that v_1, v_2 and v_3 are located at points $(0, 0), (2, 0)$, and $(1, 1)$, respectively. This allows us to draw (v_1, v_2) as a horizontal line-segment and edges (v_1, v_3) and (v_2, v_3) as quarter circular arcs. Moreover, we set $S(v_1) = \{v_1\}$, $S(v_2) = \{v_2\}$ and $S(v_3) = \{v_3\}$. According to the Schnyder coloring, we color (v_1, v_3) blue and (v_2, v_3) green. Clearly, Γ_3 fulfils Invariants 1 to 4.

Moreover, for $k = 4, \ldots, n$ assume that Γ_{k-1} has been drawn such that it fulfils Invariants 1 to 4; see Fig. 3.17a. Let $(w_1 = v_1, \ldots, w_p = v_2)$ be the sequence of vertices from left to right along C_{k-1} and let w_ℓ and w_r be the leftmost and rightmost neighbors of v_k on C_{k-1}. As in the shift-method, we translate all vertices in $\bigcup_{i=1}^{\ell} S(w_i)$ one unit to the left and all vertices in $\bigcup_{i=r}^{p} S(w_i)$ one unit to the right.

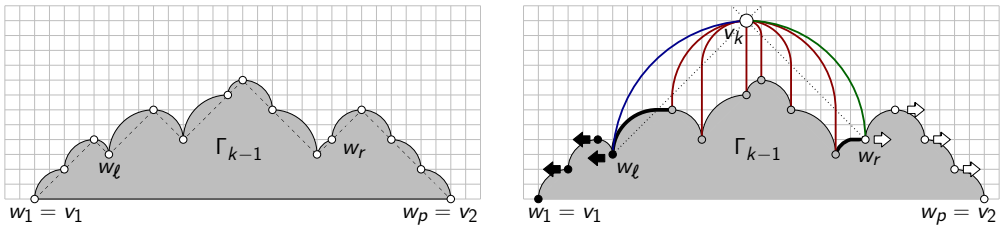


Figure 3.17: (a) Illustration of Invariant 1 maintained by the algorithm in the proof of Theorem 3.6, and, (b) illustration of the insertion of a new vertex v_k in an iteration of the algorithm in the proof of Theorem 3.6.

As a result, edges $(w_\ell, w_{\ell+1})$ and (w_{r-1}, w_r) acquire a new horizontal segment which is in accordance to Invariant 4; see the bold edges in Fig. 3.17b. Moreover, by Invariant 4, we know that all blue and green edges between different shift-sets have a horizontal segment (possibly of length zero) that can be extended to achieve a planar drawing of G_{k-1} after the shifting operation. Then, v_k is placed at the intersection of the line λ_ℓ with slope $+1$ through w_ℓ and the line λ_r of slope -1 through w_r; see the dotted lines in Fig. 3.17b. We set $S(v_k) = \{v_k\} \cup \bigcup_{i=\ell+1}^{r-1} S(w_i)$ as in the shift-method to ensure Invariant 3. Now, the new blue edge (w_ℓ, v_k) and the new green edge (w_r, v_k) can be realized as quarter circular arcs, hence Invariant 1 is satisfied. On the other hand, red edges (w_i, v_k) for $\ell < i < r$ are realized by a representation consisting of a vertical line-segment starting from w_i and ending on λ_ℓ (if w_i is to the left of v_k) or λ_r and a quarter circular arc connecting the end of the vertical line-segment with v_k. Since those new edges are intersection-free we maintain Invariant 2. We point out that the position of vertices is identical to the positioning of vertices in the shift-method. The time complexity follows analogously to the time complexity of the shift-method. □

We remark that the result from Theorem 3.6 transfers to all planar graphs since each planar graph can be triangulated to obtain a maximal planar graph. Next, we improve on Theorem 3.6 by ensuring that a certain number of edges will be drawn as a single segment at the cost of increasing the drawing area:

Theorem 3.7. *Let G be an n-vertex maximal planar graph. A planar bi-monotone SC_2-Kandinsky drawing of G in $\mathcal{O}(n^3)$ area where at least $n-1$ edges are represented by a single segment can be computed in $\mathcal{O}(n)$ time.*

Proof. For each vertex, we use the x-coordinates as computed by the algorithm in the proof of Theorem 3.6. However, we compute new y-coordinates processing the vertices in the order of the same canonical ordering $\pi = (v_1, \ldots, v_n)$. Here, we maintain the following invariants for the drawing Γ_k of G_k:

I.1 In Γ_k, all edges (v_i, v_j) of C_k for $i < j$ consist of a vertical segment (of potentially zero length) attached to v_i and a quarter circular arc unless $i = 1$

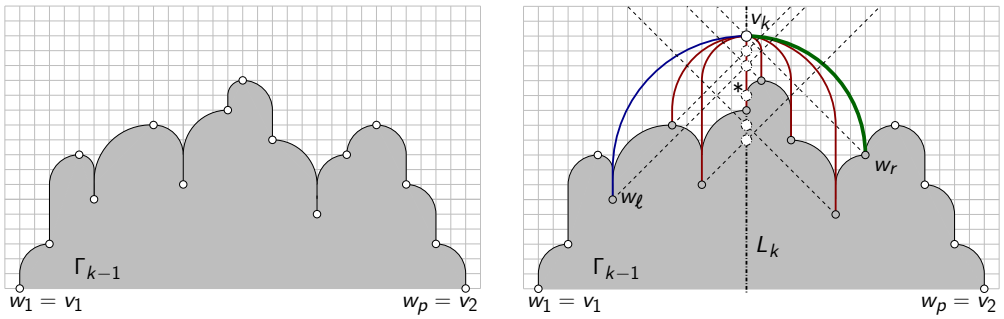


Figure 3.18: (a) Illustration of Invariant 1 maintained by the algorithm in the proof of Theorem 3.7, and, (b) illustration of the insertion of a new vertex v_k in an iteration of the algorithm in the proof of Theorem 3.7.

and $j = 2$, in which case it is a horizontal line-segment; see Fig. 3.18a. All interior edges either follow the same rules or are realized as a vertical segment.

I.2 Γ_k is planar.

I.3 G_k contains a spanning tree whose edges are realized with a single segment each in Γ_k.

Initially, we draw G_2 such that $y(v_1) = y(v_2) = 0$. This initial drawing obviously realizes all invariants. Next, assume that v_k for $k = 3, \ldots, n$ is the next vertex to be positioned and that there is a drawing Γ_{k-1} fulfilling all invariants. Further, let $(w_\ell, \ldots, w_r)$ denote the ordered sequence of neighbors from left to right along C_{k-1}. For vertex w_i with $\ell \leq i \leq r$, we draw a line ℓ_i of slope $+1$ (or -1) passing through w_i, if w_i is strictly to the left (or right, respectively) of v_k. The intersection of such a line ℓ_i with the vertical line L_k with x-coordinate $x(v_k)$ is a *candidate position* for the placement of v_k; see dashed circles in Fig. 3.18b. If for some w_i with $\ell < i < e$ it holds that $x(w_i) = x(v_k)$, we instead obtain a *trivial* candidate position at location $(x(w_i), y(w_i)+1)$; see the candidate position marked with an asterisk in Fig. 3.18b. We position v_k at the candidate position with the largest y-coordinate, that is,

$$y(v_k) = \max_{\ell \leq i \leq r} \{y(w_i) + \max\{\Delta_x(v_k, w_i), 1\}\}$$

where $\Delta_x(u, v)$ denotes the horizontal distance between vertices u and v.

Note that if $x(w_i) = x(v_k)$ for $\ell < i < r$, we realize (w_i, v_k) as a vertical segment. Let $w_{i^*} \in \{w_\ell, \ldots, w_r\}$ denote the vertex on C_{k-1} that defines the candidate position with largest x-coordinate. Since ℓ_{i^*} passes through v_k if $x(w_{i^*}) \neq x(v_k)$, (w_{i^*}, v_k) can be realized as a quarter circular arc if it is not a vertical segment. As a result, we satisfy Invariant 3. Moreover, since v_k is located above all lines ℓ_i for $\ell < i < r$, it is possible to realize all remaining edges with at most two segments;

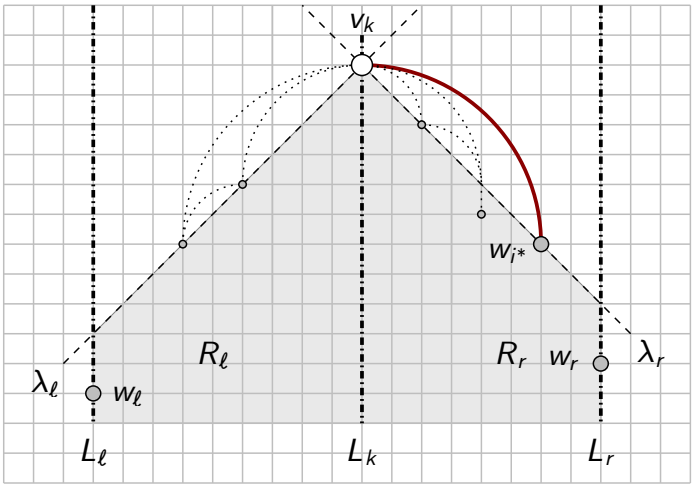


Figure 3.19: Illustration for the proof that Invariant 2 is maintained in the algorithm proving Theorem 3.7.

one vertical segment attached to w_i (possibly zero length) and one quarter circular segment attached to v_k. Hence, Invariant 1 is satisfied for Γ_k.

It remains to show that Γ_k is planar to prove Invariant 2. For this, consider vertical lines L_ℓ and L_r through w_ℓ and w_r, respectively. Moreover consider lines λ_ℓ and λ_r through v_k with slopes $+1$ and -1, respectively. By construction, all vertices w_i for $\ell \leq i \leq r$ are located in the union of regions R_ℓ and R_r, where R_ℓ (R_r) is the open region bounded by L_ℓ (L_r), λ_ℓ (λ_r) and L_k which is below λ_ℓ (λ_r), see the gray-shaded region in Fig. 3.19. Note that vertices may be located on the boundary of the region; in particular w_ℓ and w_r are located on L_ℓ and L_r, respectively, while w_i^* is either located on L_k or on one of λ_ℓ and λ_r.

Since the radii of all quarter circular arcs incident to v_k are different, those segments do not overlap. Moreover, since all vertices are located in $R_\ell \cup R_r$, v_k attaches to C_{k-1} from above, either with the end of a quarter circular arc (which has a vertical tangent at the point) or with a vertical segment. As a result, edges incident to v_k do not intersect Γ_{k-1} and planarity is guaranteed.

The time complexity follows analogously to the time complexity of the shift-method. Invariant 3 immediately implies that $n-1$ edges are drawn with a single segment. Finally, the width of the drawing remains linear in n, while the height is at most $\mathcal{O}(n^2)$ since when inserting a vertex it is placed at most $\mathcal{O}(n)$ units above the topmost vertex already present. □

We conclude the discussion of smooth orthogonal Kandinsky drawings by presenting an example run of our two SC_2-Kandinsky drawing algorithms applied on a planar triangulation on seven vertices; see Fig. 3.20. In particular, Figs. 3.20a–3.20e show the algorithm described in the proof of Theorem 3.6. Figs. 3.20f–3.20j show how the algorithm described in the proof of Theorem 3.7 computes

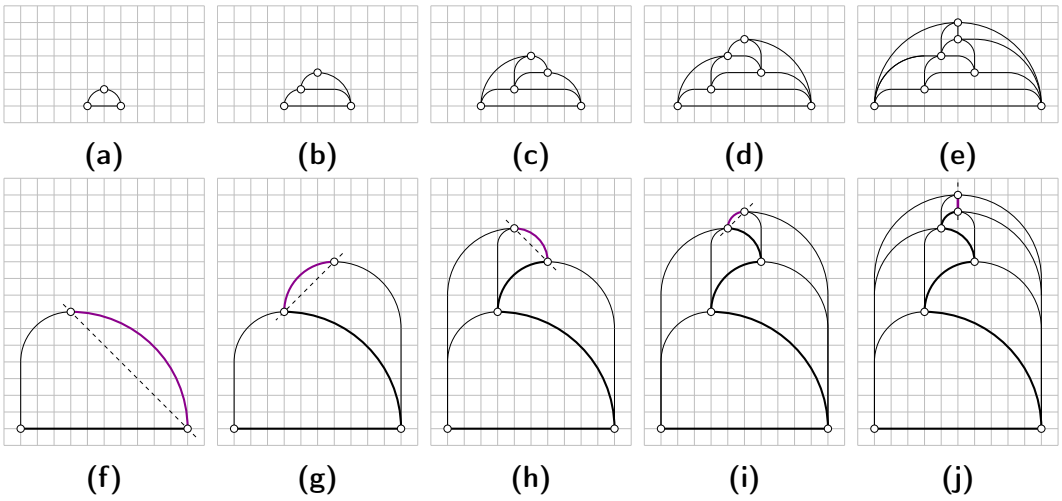


Figure 3.20: Example runs of the two SC_2-Kandinsky drawing algorithms discussed in this section with an input graph on seven vertices: (a)–(e) Algorithm in the proof of Theorem 3.6, and, (f)–(j) Algorithm in the proof of Theorem 3.7.

new y-coordinates based on the drawing in Fig. 3.20e. The bold-drawn edges in Figs. 3.20f–3.20j show the spanning tree whose edges are realized without any bends, the purple edge is the one defining the y-coordinate for the new vertex.

Finally, we shift our attention to octilinear Kandinsky drawings. Kandinsky drawings in the octilinear model have received some attention before, mostly in the context of slanted orthogonal drawings [33, 36] which are mainly used for non-planar graph drawing and have at least curve complexity three by definition. Here, we provide an algorithm that produces $8C_2$-drawings for planar graphs with large angles at every bend. While in contrast to the classic orthogonal Kandinsky drawings, we obtain smaller angles between edges at a vertex, the large angles formed by segments at bends might still prove to result in good readability. We state the analogous theorem of Theorem 3.6 for octilinear graphs:

Theorem 3.8. *Let G be an n-vertex maximal planar graph. A planar bi-monotone $8C_2$-Kandinsky drawing of G where each bend is at $\frac{3}{4}\pi$ in $\mathcal{O}(n^2)$ area can be computed in $\mathcal{O}(n)$ time.*

Proof. The proof is analogous to the proof of Theorem 3.6; it suffices to replace all quarter circular arcs by diagonal segments; see Fig. 3.21. As a result, all bends are at $\frac{3}{4}\pi$. Planarity is ensured since blue and green edges do not pass through vertices by construction. □

Again, we point out that Theorem 3.8 can be extended to all planar graphs via triangulation of the input graph.

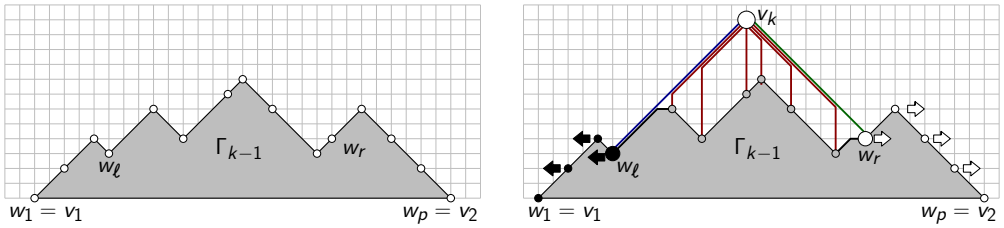


Figure 3.21: (a) Illustration of the contour condition maintained by the algorithm in the proof of Theorem 3.8, and, (b) illustration of the insertion of a new vertex v_k in an iteration of the algorithm in the proof of Theorem 3.8.

Chapter 4

Orthogonal and Smooth Orthogonal Drawings of 1-Planar Graphs

In the past, orthogonal and smooth orthogonal graph drawings have been mainly studied for planar graphs. For orthogonal drawings, this can be in part attributed to the nature of classical applications like VLSI and floor-planning, while smooth orthogonal drawings are a quite recent research direction making planar graphs the most intuitive starting point for their study.

As a result, for both drawing models, planar graphs are well understood. It is known that every connected planar graph of maximum degree four admits a planar OC_3-drawing (with the exception of the octahedron which admits a planar OC_4-drawing) [43, 129] and a planar SC_2-drawing (here, including the octahedron) [9]. For maximum degree three planar graphs, planar OC_2-drawings [117] and planar SC_1-drawings [29] can always be achieved. Moreover, in the smooth orthogonal setting, outerplane graphs admit planar SC_1-drawings in superpolynomial area [9].

On the other hand, for orthogonal and smooth orthogonal drawings of non-planar graphs, results are sparse. It is known, that every graph of maximum degree four admits both an OC_3-drawing [43] and an SC_1-drawing [29]. The drawback of both algorithms is, that they do not restrict how intersections are formed except for the fact that in the orthogonal setting they appear at right angles by construction. However, the broad study of beyond-planar graphs [72] suggests that it is worthwhile to limit how many intersections may occur on one edge. As a result, studying how a k-planar embedding can be maintained in an orthogonal or smooth orthogonal drawing of maximum degree four graphs is an interesting research question.

In this chapter[1], we begin the study of this direction of research. In particular, in Section 4.2 we prove that every 1-planar graph admits a 1-planar OC_4- and, if it is

[1]The results of this chapter also appeared in [18].

Table 4.1: Overview of our results and comparison to previous work. (*) indicates that the octahedron admits only an OC_4-drawing.

	graph class	max. deg.	curve complexity	drawing area	reference
orthogonal	general (non-planar)	4	OC_3	$n \times n$	[43]
	planar	4	OC_3 (*)	$n \times n$	[43, 129]
		3	OC_2	$n \times n$	[117]
	1-plane	4	OC_4 $\not\subseteq OC_3$	$\mathcal{O}(n) \times \mathcal{O}(n)$	Thm. 4.1 Thm. 4.2
	bicon. outer-1-plane	4	OC_3 $\not\subseteq OC_2$	$\mathcal{O}(n) \times \mathcal{O}(n)$	Thm. 4.4 Thm. 4.6
smooth orthogonal	general (non-planar)	4	SC_1	$2n \times 2n$	[29]
	planar	4	SC_2	super-poly	[9]
		3	SC_2	$\lfloor n^2/4 \rfloor \times \lfloor n/2 \rfloor$	[9]
			SC_1	super-poly	[29]
	bicon. outerplane	4	SC_1	super-poly	[9]
	bicon. 1-plane	4	SC_3	$\mathcal{O}(n) \times \mathcal{O}(n^2)$	Thm. 4.3
	bicon. outer-1-plane	4	SC_2 $\not\subseteq SC_1$	super-poly	Thm. 4.5 Thm. 4.7

biconnected, also a 1-planar SC_3-drawing preserving the input embedding. Those two results will be based on the notion of 1-planar bar visibility representations, that we discuss in Section 4.1. Further, we prove that there are indeed 1-planar graphs that require curve complexity four in the orthogonal setting. In contrast to planar graphs, this increases the curve complexity only by one in both cases, which is not achieved by drawing the planarization of the input graph with a planar drawing algorithm from the literature.

Moreover, we show in Section 4.3 that every biconnected outer-1-planar graph admits an OC_3-drawing and an SC_2-drawing, preserving the input embedding. Again, in the smooth orthogonal setting, the curve complexity increases by one in comparison to the outerplanar setting. We further show, that these two results are tight regarding the achieved curve complexity. We summarize our results and set them into context to previous results from the literature in Table 4.1.

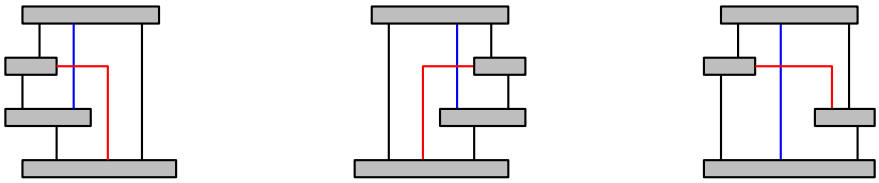


Figure 4.1: Configurations for kites in 1-planar bar visibility representations: (a) left wing, (b) right wing, and, (c) diamond.

4.1 1-Planar Bar Visibility Representations

Before we shift our attention to orthogonal and smooth orthogonal drawings, we describe how to compute a 1-*planar bar visibility representation* which will serve as an intermediate step to compute an orthogonal drawing. A similar technique has been used for orthogonal drawings of planar graphs before [152]. In a 1-planar bar visibility representation, each vertex is represented by a horizontal segment, called *bar* while each edge is represented either as a vertical segment or an *L-sequence*, that is, a sequence of one vertical and one horizontal segment such that the endpoints of the edge representation touch the bars corresponding to the vertices connected by the edge. In addition, we require horizontal segments of L-sequences to be located on top of the vertical segment and cross another edge that is represented by a vertical segment; see Fig. 4.1. We call an edge represented by an L-sequence *red* and will represent them with red color in the figures in this chapter. Similarly, we call edges intersected by red edges *blue*. Every edge will be considered as being composed of a pair of *half-edges*, which are each associated to one of the two endpoints of the edge. In particular, we split red edges at their bends to which we refer as *construction bends*. As a result, a red edge is composed of a horizontal and a vertical half-edge. Note that 1-planar bar visibility representations extend *bar visibility representations*, in which every vertex is represented by a bar and every edge by an intersection-free vertical segment. It is well-known, that every planar graph admits a bar visibility representation preserving a given input embedding [145, 151]. We outline an algorithm from [47][2] to show that every 1-planar graph admits a 1-planar bar visibility representation.

As a prepocessing step, intersections are *caged* into so-called *kites*. A kite is a K_4 induced by the endpoints of two intersecting edges such that each of the four triangles induced by the intersection and one endpoint of each of the intersecting edges is a topologically connected region in the drawing. We say an intersection is caged if the four endpoints involved in the intersection induce a kite. This can

[2]In fact, in [47], Brandenburg studies so called 1-*bar visibility representations*, in which edges are drawn as vertical segments only but are allowed to intersect one bar each. His algorithm for showing that 1-planar graphs admit such a representation in fact produces 1-planar bar visibility representations, which are more restricted.

be achieved for all intersections by augmenting G to a not necessarily simple super graph G' in which all intersections are caged and into which no planar edge can be inserted without creating a double edge [10].

After the preprocessing, the intersecting edges are removed, yielding a plane graph G_p. Then, a bar visibility representation of G_p is computed using one of the well-known algorithms from the literature [145, 151]. These algorithms make use of an st-numbering of G_p and place each bar at a y-coordinate equal to the st-number of its corresponding vertex. In the resulting bar visibility representation of G_p, quadrangular faces that bound the kites of G are drawn with one of the configurations in Fig. 4.1, called left wing, right wing and diamond. The missing edges are inserted into the kite as shown in Fig. 4.1 resulting in a 1-planar bar visibility representation of G'. Finally, the removal of the caging edges yields a 1-planar bar visibility representation of G which also respects the input embedding.[3] The resulting bar visibility representation requires $\mathcal{O}(n) \times \mathcal{O}(n)$ area.

Consider a 1-planar bar visibility representation Γ of a graph. We say that an edge e is *left, right, top* or *bottom* for a bar $\Gamma(v)$ if and only if $\Gamma(e)$ is attached to the corresponding side of $\Gamma(v)$. Observe that a red edge can be left or right edge only of one of its endpoints while it is a top edge for its second endpoint. Further, by construction, each bar has at most one left and at most one right edge. We call a bar $\Gamma(v)$ *bottom* (or *top*), if it has no top (or bottom, respectively), edges. If $\Gamma(v)$ has both bottom and top edges, we call it a *middle bar*. Let $\Gamma(v)$ be a bottom or top bar. Consider the x-coordinates of the points at which its incident edges touch $\Gamma(v)$. We say that e is *leftmost* (or *rightmost*) edge of $\Gamma(v)$ if its touching point has the smallest (or largest, respectively) x-coordinate. If $\Gamma(v)$ has a left or right edge, this edge is also leftmost or rightmost edge by construction, respectively.

4.2 1-Planar Drawings

In this section, we consider orthogonal and smooth orthogonal drawings of general 1-planar graphs of maximum degree four. First, we show that every maximum degree four 1-planar graph admits an embedding preserving OC_4-layout. Then, we prove that this result is optimal by showing that there are infinitely many graphs which do not admit an OC_3-layout. Finally, we extend our positive result to the smooth orthogonal model and show that SC_3-drawings can always be achieved for biconnected 1-planar graphs of maximum degree four.

Theorem 4.1. *Every embedded* 1*-planar graph G with n vertices and maximum degree four admits an embedding preserving OC_4-drawing on a grid of size $\mathcal{O}(n) \times$*

[3] The original description of the algorithm in [47] omits the possibility to maintain the embedding as it instead maintains simplicity of every intermediate graph. However, parallel edges can be easily simulated by a path of length two which suffices to maintain simplicity.

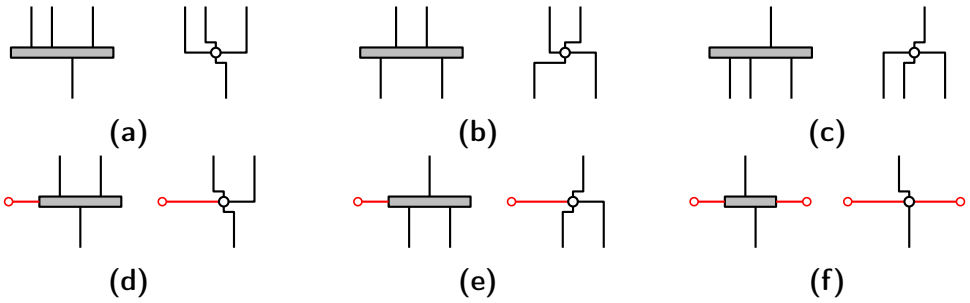


Figure 4.2: Replacement of middle bars of degree four with vertices in the case where (a)–(c) zero, (d)–(e) one, and, (f) two horizontal half-edges are present.

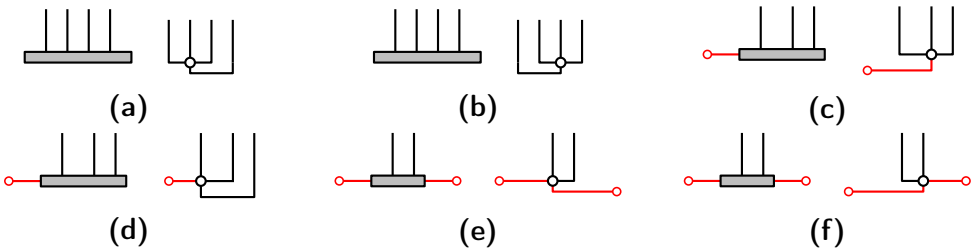


Figure 4.3: Replacement of bottom bars of degree four with vertices in the case where (a)–(b) zero, (c)–(d) one, and, (e)–(f) two horizontal half-edges are present.

$\mathcal{O}(n)$.

Proof. Our algorithm is outlined in Algorithm 4.1 and explained in the following. Without loss of generality assume that G is connected; otherwise we apply Algorithm 4.1 to every connected component. First, we compute a 1-planar bar visibility representation as described in Section 4.1.

Similar to the approach described in [152] for planar orthogonal drawings, we replace every bar by a vertex; see Figs. 4.2 and 4.3 for the degree four cases. For lower degree, we use some suitable subdrawing, and remaining cases such as for top bars are symmetric. In addition, we have to reroute some half-edges by the introduction of additional bends. We point out that some of those additional bends can be removed again: Consider two consecutive bends along an edge e with an incident topologically connected region f. If both bends are convex or concave in f, they form a so-called *U-shape*, otherwise, they form an *S-shape*. As a direct consequence of the flow model in [149], S-shapes can be removed unless an intersection appears in between the two consecutive bends. As a result, the vertex replacement introduces at most one new bend on horizontal half-edges and at most two new bends on vertical half-edges; see Fig. 4.2 and 4.3. We call a half-edge *extreme* if it is a horizontal segment and got one additional bend or if it is vertical and got two additional bends creating a U-shape.

We claim that edges can be routed such that no edge is composed of two extreme half-edges which contribute two bends each (where horizontal extreme

Input: An embedded 1-planar graph $G = (V, E)$ and an edge $(s, t) \in E$
Output: An OC_4-layout Γ_o of G

1 $\pi = (s = v_1, \ldots, t = v_n) \leftarrow st$-numbering(G);
2 $\Gamma_b \leftarrow$ 1-planar-bar-visibility(G, π);
3 Let V_B denote the vertices realized with top and bottom bars in Γ_b;
4 $\Gamma_o \leftarrow \Gamma_b$;
5 **for** $v \in V \setminus V_B$ **do**
6 Replace bar $\Gamma_o(v)$ with a node-link representation according to Fig. 4.2;
7 Remove S-shapes on half-edges incident to $\Gamma_o(v)$;
8 **end**
9 Let V_E denote the set of leftmost and rightmost edges of vertices in V_B and let $H = (V_B \dot\cup V_E, E_H)$ such that $(v_b, v_e) \in E_H$ if and only if v_e is leftmost or rightmost edge of v_b;
10 $M_H \leftarrow$ maximum-matching(H);
11 **for** $v \in V_B$ **do**
12 Let e be the edge matched to v in M_H;
13 Replace bar $\Gamma_o(v)$ with a node-link representation according to Fig. 4.3 such that the half-edge of e incident to v receives two bends;
14 **end**
15 **return** Γ_o;

Algorithm 4.1: Algorithm for computing an OC_4-layout. Note that Line 1 assumes G to be biconnected. This property can be easily achieved by the insertion of dummy edges if it is not given.

half-edges contribute the additional bend plus their construction bend). This suffices to show that the resulting drawing is an OC_4-drawing. We observe that half-edges can only be extreme if they are a leftmost or rightmost edge of a bottom or top bar. Note that it can be chosen freely which one of the leftmost and rightmost half-edge will become an extreme edge in either case; see the pairs of Figs. 4.3a–4.3b and 4.3c–4.3d and 4.3e–4.3f. Hence consider a bipartite graph H whose vertex set is composed of the set of top and bottom bars V_B and the set of leftmost and rightmost edges of such bars V_E. Further, a bar $b \in V_B$ is connected to an edge $e \in V_E$ if and only if e is incident to b. As a result, all vertices in V_B have degree exactly two while all vertices in V_E have degree at most two. Hence, H is a union of disjoint paths and cycles such that each path or cycle h in H contains at least as many vertices of V_B as h contains vertices of V_E. Consequently, there is a matching M_H covering V_B. If $(b, e) \in M_H$ for $b \in V_B$ and $e \in V_E$, we define the half-edge of e incident to b to be its extreme half-edge. This assigns exactly one extreme half-edge to every top and bottom bar while only one half-edge per edge becomes extreme. This concludes the proof. □

Next we prove a corresponding lower bound for the required edge complexity.

Figure 4.4: Illustrations for the proof of Theorem 4.2: (a) Component graph C, and, (b) Graph composed of a chain of components isomorphic to C.

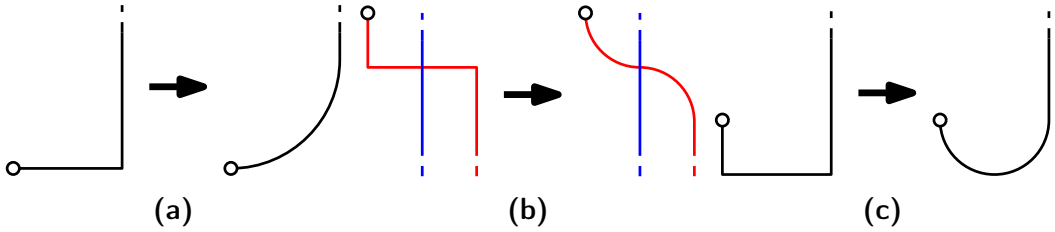


Figure 4.5: Conversion of half-edges from an orthogonal layout to a smooth orthogonal layout. (a) L-shaped half-edge, (b)-intersect S-shaped half-edge, and, (c) U-shaped half-edge.

Theorem 4.2. *For every $k \geq 2$, there exists a biconnected 1-planar graph G_k with maximum degree four and fixed embedding $\mathcal{E}_k$ on $9k$ vertices that admits no 1-planar OC_3-drawing realizing $\mathcal{E}_k$. In addition, k edges of complexity at least four are required in any embedding preserving OC_4-drawing.*

Proof. The central component of our proof is an embedded 1-planar graph C on 9 vertices shown in Fig. 4.4a. C contains a triangle $T = (a, b, c)$ which must be embedded such that all ports of vertices a, b and c point inside T. As a result, T requires at least 7 bends and one of the edges forming it must have at least three bends; see edge (b, c) in Fig, 4.4a. The outer face of C is formed by a planar triangle (s, x, t). As a result, k copies $C_1, \ldots, C_k$ of C can be arranged in a chain by connecting vertex t from copy C_i with vertex s from copy $C_{(i+1) \bmod k}$ for $1 \leq i \leq k$. This concludes the proof. □

In the next theorem, we reconsider the result from Theorem 4.1 in order to compute an SC_3-drawing of a 1-planar input graph.

Theorem 4.3. *Every embedded biconnected 1-planar graph G with n vertices and maximum degree four admits an embedding preserving SC_3-drawing on a grid of size $\mathcal{O}(n) \times \mathcal{O}(n^2)$.*

Proof. We first compute an OC_4-drawing Γ_o with Algorithm 4.1. Recall that Algorithm 4.1 treats edges as being composed of half-edges where construction

Figure 4.6: (a) Proof that u is below u', and, (b) proof that e' (blue) is rightmost outgoing edge of u' for certain pairs of intersecting edges (u, v) and (u', v') with bottom endpoints u and u'.

bends of red edges are assigned to their top half-edge. As a result, each half-edge is either a single vertical segment, an *L-shape* (that is, half-edges with one bend), an S-shape that is intersected by another edge (between both bends) or a U-shape. In addition, each edge can have at most one half-edge that is S-shaped and intersected or U-shaped. We convert Γ_o into a smooth orthogonal drawing Γ_s by first stretching Γ_o along appropriate cuts and by then replacing L-shaped half-edges by a sequence of a quarter circular arc and a vertical segment (see Fig. 4.5a), intersected S-shaped half-edges by a sequence of two quarter circular arcs and a vertical segment (see Fig. 4.5b), and, U-shaped half-edges by a half circular arc followed by a vertical segment (see Fig. 4.5c). Since the two vertical segments of consecutive half-edges are adjacent, they can be merged into a single segment. As a result, an SC_3-drawing is obtained, except for the special case, where an edge is composed of an intersected S-shaped half-edge and an L-shaped half-edge. Note that Algorithm 4.1 ensures that edges are never composed of an intersected S-shaped half-edge and a U-shaped half-edge.

Consider an edge $e = (u, v)$ with source u and target v that has an intersected S-shaped half-edge h_S incident to v and L-shaped half-edge h_L incident to u. If the bend of h_L and the construction bend of e (which is assigned to h_S) form an S-shape, we can remove both bends [149]. Otherwise, the two bends form a U-shape. In this case, consider edge $e' = (u', v')$ that intersects e. Assume w.l.o.g. that e is attached to u at the east port while u' is located left of e; see Fig. 4.6. We first make two observations: First, u is located below u', and, second, that e' is the rightmost outgoing edge of u'. We show both statements by contradiction.

First, assume, u' has a smaller st-number than u. Since the north port was not used by e, vertex u must have another outgoing edge e_N attached to the north port of u which is part of a path P leading to the global sink t since G is biconnected. Then, e_N's target is located inside a region that is bounded from above by the two intersecting edges e and e' and hence P cannot exist; see Fig. 4.6a. This contradicts our initial assumption and we conclude that u is located below u'.

Second, assume that u' has another outgoing edge e_R that leaves u' to the

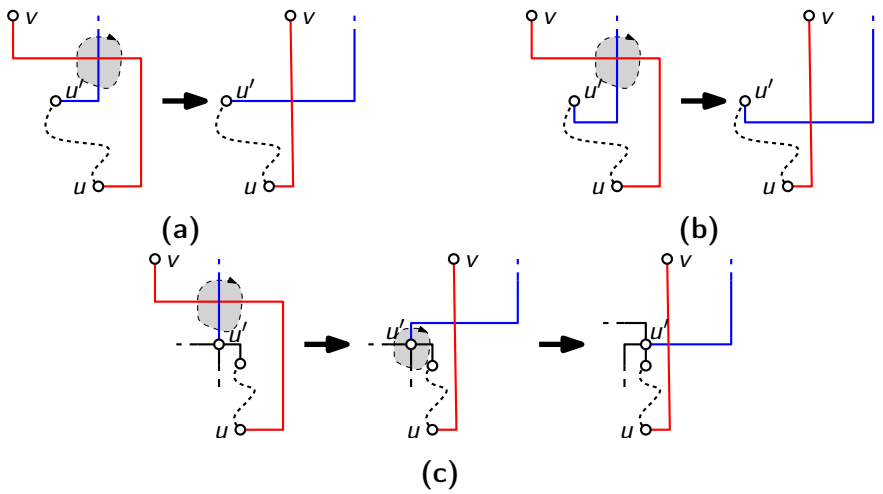


Figure 4.7: Removal of intersected S-shaped half-edges from a red edge $e = (u, v)$ intersected by an edge e' (blue) with source u'. The case where e' has (a) one bend, (b) two bends, and, (c) no bend at the half-edge connected to its source. In either case local rotations around the intersection of e and e' or the source of e' (see dashed circles) remove the intersected S-shape.

right of e'. Then, the target of e_R is located in a region that is bounded from above by the two intersected edges e and e'; see Fig. 4.6b. This contradicts the biconnectivity of G and hence e' is the rightmost outgoing edge of u'.

We perform a rotation around the intersection of e and e'; see the dashed arrows in Fig. 4.7. As a result, h_S becomes a straight-line segment. Consider the half-edge h' incident to the source u' of e'. If e' uses the east or south port of u', h' has a horizontal segment which is initially intersection-free and a vertical segment which intersects e. As a result of the rotation, the intersection with e' now occurs on the horizontal segment while the shape of h' remains the same; see Figs. 4.7a and 4.7b. Hence, the curve complexity of e' does not increase in those cases. If h' uses the north port of u', we instead let it use the west port if this port is free. Otherwise, h' becomes an intersected S-shaped half-edge while the edge e_E attached to the east port of e' connects u' to a vertex with lower st-number; see the middle drawing in Fig. 4.5c. We may assume w.l.o.g. that u' is also incident to edges e_S and e_W incident to the south and west ports of u', respectively, such that e_S connects u' with another vertex with lower st-number. In this case, we also rotate around u' adding another bend on all incident edges. In particular, this will make the half-edge of e_E incident to u' become a straight-line segment, while it adds a bend to both the half-edges of e_W and e_S incident to u'; see the right drawing in Fig. 4.5c. However, none of those additional bends can result in a new intersected S-shape, in particular, if e_S was such an intersected S-shape before, a new S-shape is formed whose bends can be removed. The half-edge of e_W incident to u' will become either an S-shape (which can be removed) if e_W connects u' to

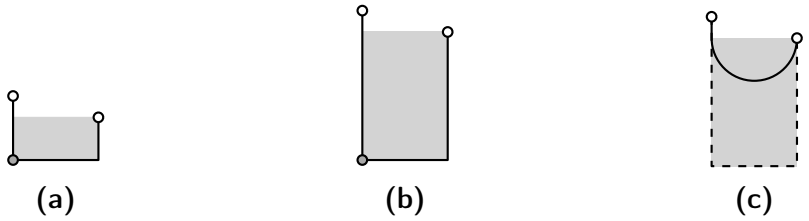


Figure 4.8: Transformation of U-shapes from an orthogonal to a smooth orthogonal drawing. (a) Initial U-shape with dummy vertex, (b) stretched U-shape, and, (c) replacement by a half circular arc.

a vertex with larger st-number, or it will become a U-shaped half-edge otherwise.

After this preprocessing, the intermediate orthogonal layout has certain properties: First, the area bounded by U-shaped half-edges is free of vertices; see the gray area in Fig. 4.8a. In addition, each edge that has an intersected S-shaped half-edge has another straight-line half-edge. Moreover, w.l.o.g., each vertex is on a unique level as otherwise horizontal straight-line edges can be replaced by an S-shaped edge. We now split each U-shaped half-edge with a dummy vertex into an edge represented by a vertical straight-line segment and an L-shaped half-edge; see the gray vertex in Fig. 4.8a. Let $v_1, \ldots, v_{n'}$ denote the vertices of the graph obtained after introducing all dummy vertices ordered in ascending vertical order. Further, let $\Delta_x^{\uparrow}(v_i)$ denote the largest horizontal distance between v_i and a bend on its incident L-shaped half-edges that connect v_i to a vertex v_j with $j > i$. Analogously, let $\Delta_x^{\downarrow}(v_i)$ denote the largest horizontal distance between v_i and a bend on its incident L-shaped half-edges and red intersected S-shaped half-edges that connect v_i to a vertex v_j with $j < i$. We stretch the drawing by increasing the y-coordinate of all v_j with $j \geq i$ by $\Delta_x^{\downarrow}(v_i)$ and by increasing the y-coordinate of all v_j for $j > i$ by $\Delta_x^{\uparrow}(v_i)$. Note that this corresponds to stretching along horizontal cuts slightly below and above v_i through the drawing, respectively. It is noteworthy, that the area enclosed by U-shaped half-edges remains free of vertices; see Fig. 4.8b.

After the stretching procedure, the dummy vertices introduced at U-shaped half-edges are removed. We then replace half-edges as described before and as illustrated in Fig. 4.5. In particular, the half-circles replacing U-shaped half-edges are entirely located inside the vertex-free area; see Fig. 4.8c.

Since the orthogonal drawing Γ_o had $\mathcal{O}(n) \times \mathcal{O}(n)$ area and since for every vertex we increase the height by at most twice the length of the longest horizontal segment in Γ_o, the smooth orthogonal drawing Γ_s requires $\mathcal{O}(n) \times \mathcal{O}(n^2)$ area. □

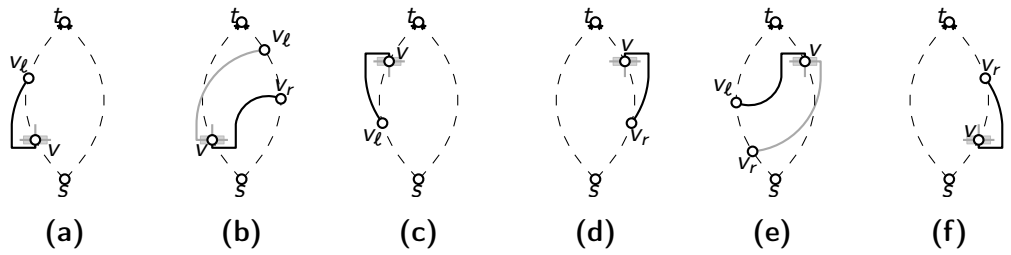


Figure 4.9: Replacement of bottom and top bars in the proof of Theorem 4.4.

4.3 Outer-1-Planar Drawings

In this section, we shift our attention to orthogonal and smooth orthogonal drawings of outer-1-planar graphs. It is noteworthy that such graphs are in fact planar graphs [20], hence, they clearly admit both a planar OC_3- and an SC_2-drawing while an OC_2- or an SC_1-drawing may not be achievable. Here, we show a stronger result, namely, that the same curve complexity bounds apply when the outer-1-planar embedding must be preserved. We start by adapting our result from Theorem 4.1 for the outer-1-planar case.

Theorem 4.4. *Every embedded biconnected outer-1-planar graph G with n vertices and maximum degree four admits an embedding preserving OC_3-drawing on a grid of size $\mathcal{O}(n) \times \mathcal{O}(n)$.*

Proof. We adapt Algorithm 4.1 to yield the given result. In particular, we make three adaptations: First, we use a specific st-ordering (cf. line 1 in Algorithm 4.1) that will make it easier for us to control which edges have at least three bends. Second, we will specify based on the st-numbering, which half-edges incident to bottom bars receive two bends (in contrast to the specification by a matching in line 13 in Algorithm 4.1). This specification of half-edges with two bends may initially result even in edges with four bends, which are avoided by Algorithm 4.1. Finally, we remove S-shapes from edges with three or four bends to obtain an OC_3-layout which we will show is always possible.

In order to compute the st-ordering, we first augment G such that all intersections are caged. The resulting graph G' is bounded by a planar cycle C. Let s and t be two vertices of G and let S_ℓ and S_r denote the sequences of vertices encountered following the clockwise and counterclockwise path, respectively, from s to t along C. We choose as st-ordering the sequence s, S_ℓ, S_r, t.

Based on the computed st-ordering, we proceed normally with Algorithm 4.1 until we have to replace top and bottom bars of degree four with a node-link representation (cf. line 13 in Algorithm 4.1). Let v be a top or bottom bar of degree four and let $e_\ell = (v, v_\ell)$ and $e_r = (v, v_r)$ be its leftmost and rightmost incident edge, respectively. We distinguish the following cases for such vertices:

1) $v \in S_\ell \cup \{s\}$ **and** v **is bottom bar.** If $v_\ell \in S_\ell$, we attach e_ℓ to the south port of v; see Fig. 4.9a. Note that e_ℓ cannot be a red edge connecting v with a vertex below v, as otherwise, it is intersected by a blue edge which prevents v from being part of S_ℓ. Otherwise, $v_r \in S_r$ and we attach e_r to the south port of v; see Fig. 4.9b.

2) $v \in S_\ell \cup \{s\}$ **and** v **is top bar.** Then, all neighbors of v appear below v and we attach e_ℓ to the north port of v; see Fig. 4.9c.

3) $v \in S_r \cup \{t\}$ **and** v **is top bar.** If $v_r \in S_r$, we attach e_r to the north port of v; see Fig. 4.9d. Otherwise, $v_\ell \in S_\ell$ and we attach e_ℓ to the north port of v; see Fig. 4.9e.

4) $v \in S_r \cup \{t\}$ **and** v **is bottom bar.** Then, all neighbors of v connected with planar edges appear above v and we attach e_r to the south port of v; see Fig. 4.9f. Note that e_r cannot be a red edge connecting v with a vertex below v, as otherwise, it is intersected by a blue edge which prevents v from being part of S_r.

It remains to analyze the resulting drawing focusing on edges with at least three bends. We first show that if edge $e = (u, v)$ with u below v has at least three bends, two of these bends are vertically aligned and form an S-shape. We consider three cases based on whether the endpoints of edge e belong to $S_\ell \cup \{s\}$ or $S_r \cup \{t\}$:

1) $u, v \in S_\ell \cup \{s\}$**.** Since e has two bends, it either uses the south port of u or the north port of v. First, assume that e is connected to the south port of u. This is the case, when u is a bottom bar of degree four while e is the leftmost edge of u. At least two of the remaining neighbors of u are placed above u, the exception is the rightmost neighbor which may be connected to u by a red edge. Hence, the remaining neighbors of v that are connected to the bottom side of the bar of v are located between u and v and e is the rightmost edge at the bottom of $b(v)$; see Fig. 4.10a. As a result, e can only use the south or east port of v even when v is a top bar as then the leftmost edge will use its north port. If e uses the south port of v, it has only two bends, otherwise, two vertically aligned bends form an S-shape, see Fig. 4.10b.

   Second, assume that e uses the north port of v. This is the case if v is a top bar of degree four with e being its leftmost edge. It follows immediately that all other neighbors of v are drawn below u. Similar to the argument in the first case, we conclude that e is attached to the north or the east port of u. Again, we only obtain three bends when e uses the east port of u, in which case two vertically aligned bends create an S-shape as required; see Fig. 4.10c.

2) $u, v \in S_r \cup \{t\}$**.** This case is symmetric to the previous case in which $u, v \in S_\ell \cup \{s\}$; see Figs. 4.10d–4.10f.

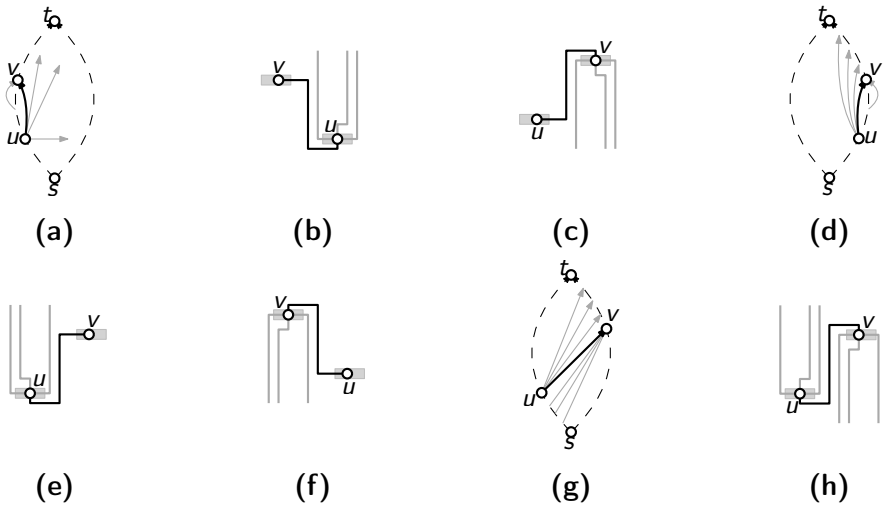


Figure 4.10: Figures for proving that edges of an outer-1-planar graph with three or four bends have two vertically aligned bends forming an S-shape when bars are replaced according to the rules described in the proof of Theorem 4.4.

3) $u \in S_\ell \cup \{s\}$ **and** $v \in S_r \cup \{t\}$**.** First, assume that e uses the south port of u. If this is the case, u is a bottom bar of degree four and e the rightmost edge attached to u. As a consequence, v cannot be connected to vertices in S_ℓ which are placed above u and e is the leftmost edge attached to the bottom of the bar representing v; see Fig. 4.10g. Therefore, e is not attached to v at the east port of v. Analogously, if e uses the north port of v, it cannot be attached to the west port of u. We conclude that e has at most four bends while if it has at least three bends, two of them are vertically aligned and form an S-shape; see Fig. 4.10h for an illustration of the case where e has four bends.

We established now, that in all cases where an edge e has at least three bends, two of them are vertically aligned and form an S-shape. If e is planar or a red edge (which is always intersected at a horizontal segment), the S-shape can be eliminated [149] and we obtain a representation of e with at most two bends. Hence, it remains to consider the case, where e is a blue edge such that the vertical segment that is part of the S-shape is intersected by a red edge $e' = (u', v')$. Recall that the algorithm described in Section 4.1 ensures that the the blue edge e is always incident to the topmost bar involved in the intersection configuration, i.e., v is located above v'; see also Fig. 4.1. We distinguish based on the type of configuration that contains the intersection:

1) e **and** e' **form a left wing configuration.** Since u must be located on the outer face, it belongs to $S_\ell \cup \{s\}$. Since u' is located below u, it also is part of $S_\ell \cup \{s\}$. Finally, by construction v' is located left of e and it follows that also $v' \in S_\ell \cup \{s\}$; see Fig. 4.11a. Since e and e' intersect, u cannot have

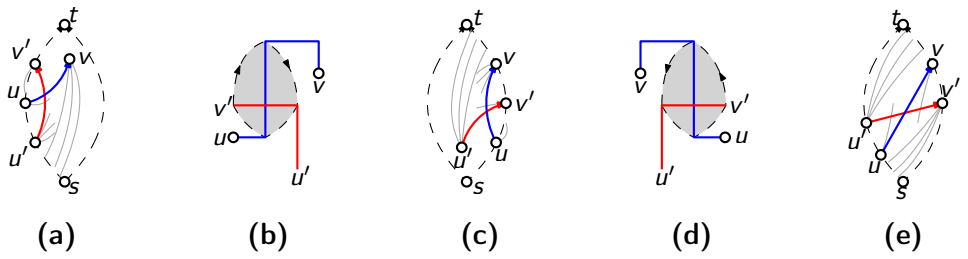


Figure 4.11: (a)–(d) A blue edge $e = (u, v)$ with three and more bends involved in a wing configuration can be simplified as removing the S-shape from e will create a new S-shape on edge $e' = (u', v')$. (e) A blue edge involved in a diamond configuration has at most two bends.

another edge right of e to a vertex above u or a neighbor in $S_r \cup \{t\}$ that is not v. Hence, e will not use the south or west port of u. Since e has at least three bends, e is attached to the east port of u and to the north port of v; see Fig. 4.11b. Since e is attached to the north port of v, v must be a degree four top bar. Moreover, its rightmost edge e_r must be incident to a vertex in $S_\ell \cup \{s\}$, since otherwise e_r would be connected to the north port of v. It follows that u' has no edge to a vertex above u' that is attached to u' right of edge e' while v is the only potential neighbor of u' belonging to $S_r \cup \{t\}$. Hence, e' can only be attached to the north or west port of u' while it is attached to the south or east port of v'. It follows that the construction bend of e' was not removed before as it was not creating any S-shape. As a result, we can apply a rotation around the intersection of e and e'; see Fig. 4.11b. This removes the S-shape on edge e and adds two bends to e', however, one of those bends creates an S-shape with the construction bend of e' so that both bends can be removed. In other words, this moves the construction bend of e' to the other side of e by letting the two edges intersect at a vertical segment of e' without changing the shape of e'. Hence, afterwards both e and e' have at most two bends each.

2) e **and** e' **form a right wing configuration.** This case is symmetric to the previous case, as indicated by Figs. 4.11c–4.11d.

3) e **and** e' **form a diamond configuration.** Then, u appears below u' while v appears above v'. Hence, edges e and e' only intersect if $u, u' \in S_\ell \cup \{s\}$ and $v, v' \in S_r \cup \{t\}$. Similar to the wing configuration cases, it follows that e is rightmost edge of u leading to vertices above u except for a possible edge (u, v') whereas e is also the leftmost edge of v leading to vertices below v except for a possible edge (u', v). Hence, e will use the north or east port of u and the south or west port of v and has at most two bends as required.

We conclude that we can always reduce the number of bends to two bends per edge which results in an embedding preserving OC_3-drawing of G as required. □

Figure 4.12: (a) Caging of cut vertices of G_p in the auxiliary graph G', and, (b) traversal (red) of a cycle around an in-dummy x_i in the weak dual of G'. Face f_4 is first inserted as a triangle with virtual reference edge (v, v').

Next, we discuss smooth orthogonal drawings of outer-1-planar graphs. In contrast to the general planar case discussed in Section 4.2, we cannot simply convert the OC_3-drawing obtained by Theorem 4.4. Instead, we will modify an algorithm for outerplanar SC_1-drawings [9] to achieve SC_2-drawings in the outer-1-planar case. The SC_1-drawing algorithm by Alam et al. first computes the weak dual tree T of the input outerplanar graph and then traverses T face by face in a BFS traversal. When face f is encountered, a *reference edge* shared with its parent face in T is already drawn. Then, the remaining vertices of f are inserted such that f is outerplanar.

Theorem 4.5. *Every embedded biconnected outer-1-planar graph G with n vertices and maximum degree four admits an embedding preserving SC_2-drawing.*

Proof. We follow the same structure as the algorithm by Alam et al. [9]. Since G is 1-planar, the weak dual graph is not well defined, hence, we first compute a biconnected auxiliary graph G' to define a suitable order in which we will insert vertices. Consider the planarization G_p of G. A dummy vertex x that replaces an intersection in the embedding of G is of one of the following three types:

(i) x is a cut vertex of G_p. Then we call x a *dummy-cut*.

(ii) x is not a cut vertex and located on the outer face of G_p. Then we call x an *out-dummy*. Note that exactly two clockwise consecutive incident edges of x are incident to the outer face.

(iii) x is not a cut vertex and not located on the outer face of G_p. Then we call x an *in-dummy*.

We obtain G' from G_p by dealing with dummy-cuts x_c as follows: Let (u, v') and (v, u') be the pair of edges in G whose intersection has been replaced by x_c. Then, we create the 4-cycle (u, v, v', u') around x_c by inserting *virtual edges* if the corresponding edges do not already exist and remove x_c from G_p; see Fig. 4.12a. We call face (u, v, v', u') of G_p a *cut-face* while faces that are not a cut-face are

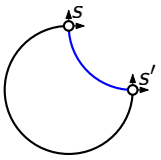


Figure 4.13: Initial drawing for our SC_2 drawing algorithm.

called a *normal face*. All other vertices and edges in G' remain as in G_p. Clearly, G' is biconnected and has maximum degree four. Consider an edge e of G'. Edge e can be either a planar edge in G, a *half-edge*, that is an edge incident to an in-dummy or out-dummy, or a virtual edge caging a dummy-cut. Moreover, the weak dual of G' is almost a tree, the only exception is cycles of length four that may occur around in-dummies x_i; see Fig. 4.12b. Note that since G_p is obtained from an outer-1-planar embedding of G, the cycles of length four are edge disjoint. We order the faces of G' using a leftmost BFS of its weak dual starting from some face f_0. In particular, the cycles of faces in G' surrounding an in-dummy x_i form two directed paths of length two in the orientation obtained by the BFS traversal, say (f_1, f_2, f_4) and (f_1, f_3, f_4); see Fig. 4.12b. Faces f_2 and f_3 will be processed consecutively and we call them a *facial pair*. We will process faces f_1, f_2 and f_3 normally, however, instead of immediately drawing f_4, we first insert the triangle (x_i, u', v') where u' and v' are the two vertices on the boundary of f_4 that x_i is incident to. We point out that (u', v') may not exist in G, in this case it will serve as a *virtual reference edge* for drawing f_4. In either case, (u', v') will be drawn as a convex quarter circular arc.

We first define how to find a suitable face f_0 for the start of the BFS. To do so, we claim that there is an edge $e_s = (s, s')$ on the outer face of G_p that is either a planar edge or a half-edge incident to some out-dummy. To see this, consider a leaf-component C in a BC-tree $\mathcal{T}$[4] decomposition of G_p. If C is also the root of the BC-tree, it contains no dummy-cut and the claimed property is trivially fulfilled. Otherwise, C has exactly one dummy-cut with degree two and hence at least two more vertices u and v. As there is no second dummy-cut and since C is biconnected, there is path between u and v that consists entirely of planar edges and half-edges incident to out-dummies as claimed. We choose as f_0 the inner face incident to e_s.

As an intial drawing, we draw edge e_s as a three-quarter circular arc. In addition, we add the virtual reference edge (s, s') realized as a concave quarter circular arc; see Fig. 4.13. Based on the BFS traversal of the weak dual of G' starting from f_0, we incrementally construct G_p. We point out a significant difference to the algorithm for outerplanar graphs: When a face f of G' is first encountered, the already drawn *reference edge* e_r may either belong to G_p or it may be a virtual

[4]A BC-tree decomposition $\mathcal{T}$ of a graph G has a vertex for every biconnected component and for every cut-vertex and an edge (C, x) if x is a cut vertex of a biconnected component C.

edge that does not belong to G_p. In the latter case, when we draw f, we remove the reference edge e_r and reuse the ports that were occupied by e_r. In particular, this is the case for the first face f_0, as initially we only draw e_s. Before we define suitable invariants, we introduce the notion of a so-called *side-arc*. A side-arc is an edge that uses the north port of one of its endpoints and the west port of its other endpoint or the west port of one of its endpoints and the south port of its other endpoint. This notion will become clear when we encounter such a case. During each step we maintain the following invariants.

I.1 Half-edges incident to an out-dummy x are drawn with a single segment unless they are a side-arc incident to the south or west port of x. In the latter case, they are composed of two segments, a straight-line segment incident to x and a quarter circular arc.

I.2 A virtual reference edge for a normal face is always drawn as a quarter-circular arc which can be convex or concave.

I.3 A non-virtual reference edge for a cut-face is composed of a single segment which can be a convex or concave quarter circular arc or a horizontal or vertical segment.

I.4 If a virtual reference edge for a cut-face is a side-arc, it is drawn with two segments. Namely, with a quarter-circular arc incident to the north or west port of one of its endpoint and a straight-line segment incident to its second endpoint.

I.5 If a non-virtual edge is drawn as a convex quarter-circular arc, its two endpoints are not dummy vertices and have remaining degree at most one.

I.6 Planar edges and edges whose intersections produce in-dummies and dummy-cuts are drawn with at most two segments.

I.7 Let (u, v) be the reference edge of a face f such that u is to the right or below v. Let L_u and L_v denote the lines of slope $+1$ passing through u and v, respectively. Further, let $L_{u,v}$ be the semi-strip in between L_u and L_v, where L_u and L_v are not included, which is bounded by the diagonal through u and v. Also, let w denote the width of $L_{u,v}$, i.e., the distance between L_u and L_v, and let $L^t_{u,v}$ and $L^b_{u,v}$ denote the semi-strip of width $0.42w$ above and below $L_{u,v}$, respectively, which are bounded by the straight-line through u and v. In addition, let $L^s_{u,v}$ denote the semi-strip of width $0.16w$ in the center of $L_{u,v}$. Then:

- if f is using neither the east port of u nor the north port of v, then, if it is not a virtual reference of a cut face, f is located entirely in $L_{u,v}$, otherwise, f is located entirely in $L^s_{u,v}$.

- if f is using the east port of u but not the north port of v, f is located entirely in $L_{u,v} \cup L^b_{u,v}$.
- if f is using the north port of v but not the east port of u, f is located entirely in $L_{u,v} \cup L^t_{u,v}$.
- if f is using both the east port of u and the north port of v, f is located entirely in $L_{u,v} \cup L^t_{u,v} \cup L^b_{u,v}$.

Moreover, if one of the side arcs of such a face is virtual reference edge for a cut-face, the cut-face is also located in the corresponding semi-strip. Note that we will not discuss Invariant 7 in the case analysis; instead, we visualize $L_{u,v}$ and $L^s_{u,v}$ in gray and $L^t_{u,v}$ and $L^b_{u,v}$ in dark-gray in the figures of this section. For better readability, the width of strip $L^s_{u,v}$ is not up to scale in the illustrations.

I.8 All free ports of vertices are located on the outer face. We illustrate this in the figures in this section by small arrows connected to newly placed vertices. Furthermore, we use ports from reference edges such that no free port is located inside the newly drawn face which is easily visible from the figures.

Let f be the next face to be drawn. Consider the type of f:

(i) **f is a normal face that does not belong to a facial pair.** Based on the shape of the reference edge (u, v) and the number of vertices to be inserted, we choose a suitable case from Fig. 4.14. Note that if we create a side-arc as for instance, if (u, v) is a horizontal segment and f is a triangle, we can decide between two different drawings; see Figs. 4.14a and 4.14i. We make our decision based on whether the specific side-arc will be a virtual reference edge for a cut-face; if this is the case, we realize the side arc with two segments (see the red edges in Fig. 4.14), otherwise, we chose the drawing where it is realized with one segment. This guarantees Invariant 4 and since such edges are not half-edges incident to out-dummies also Invariant 1. If edge (u, v) is a virtual reference edge it is drawn as a quarter circular arc by Invariant 2 and we choose the corresponding drawing from Fig. 4.15. Note that we reuse the ports of the reference edge while we do not create any side-arcs. In each case, Invariants 2, 5 are trivially fulfilled as no such edges are inserted. Finally, the planar edges fulfill the requirements of Invariants 3 and 6.

(ii) **f is a normal face belonging to a facial pair.** Recall that a facial pair is formed by two faces surrounding an in-dummy x_i; see Fig. 4.12b. In particular, the four faces surrounding x_i are f_1, f_2, f_3 and f_4, where we have already realized a drawing of face f_1, which placed x_i and its two incident half-edges (x_i, u) and (x_i, v) while $f = f_2$. Note that half-edges (x_i, u) and (x_i, v) are either a straight-line segment or a concave quarter-circular arc as they where constructed by one of the constructions from Fig. 4.14.

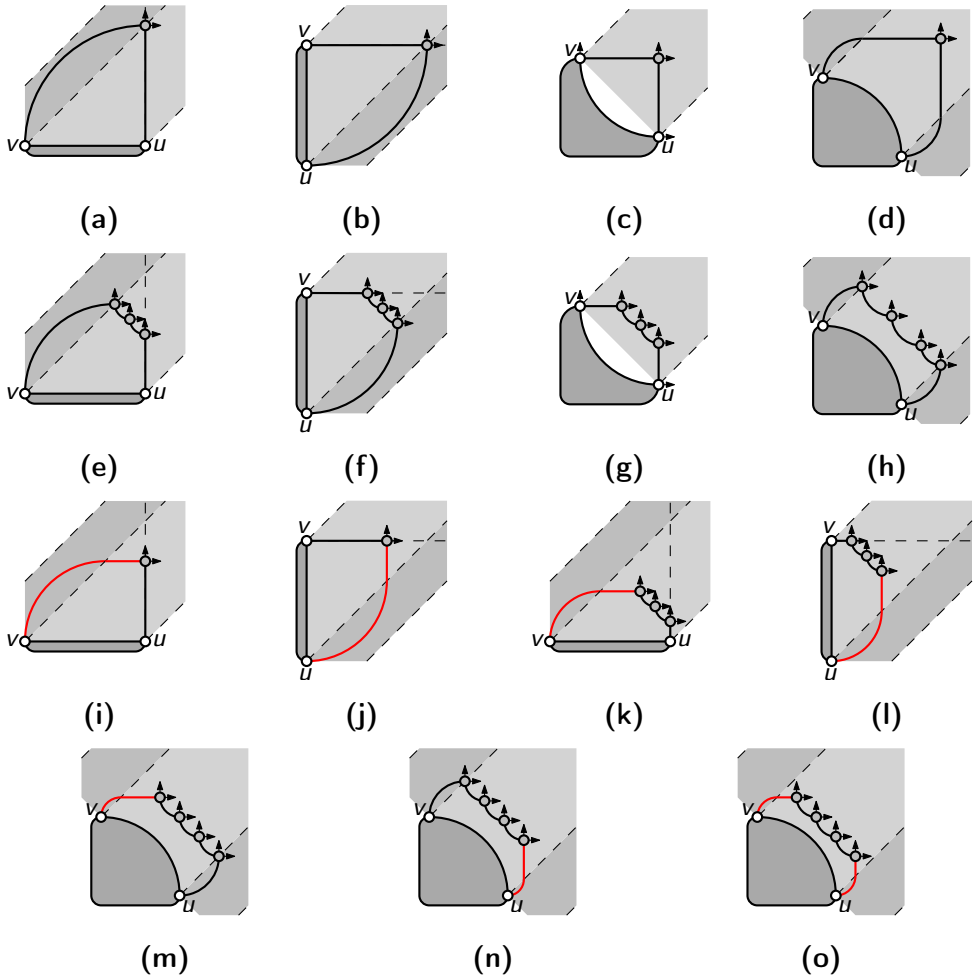


Figure 4.14: Drawing of a normal face with non-virtual reference edge (u, v). (a)–(h) Standard cases. (i)–(o) Special cases, where at least one of the side-arcs is a virtual reference edge for a cut-face (red). The cases in subfigures (a)–(c) and (e)–(g) are identical to the corresponding cases in the algorithm for outerplanar graphs [9].

Figure 4.15: Drawing of a normal face with virtual reference edge (u, v).

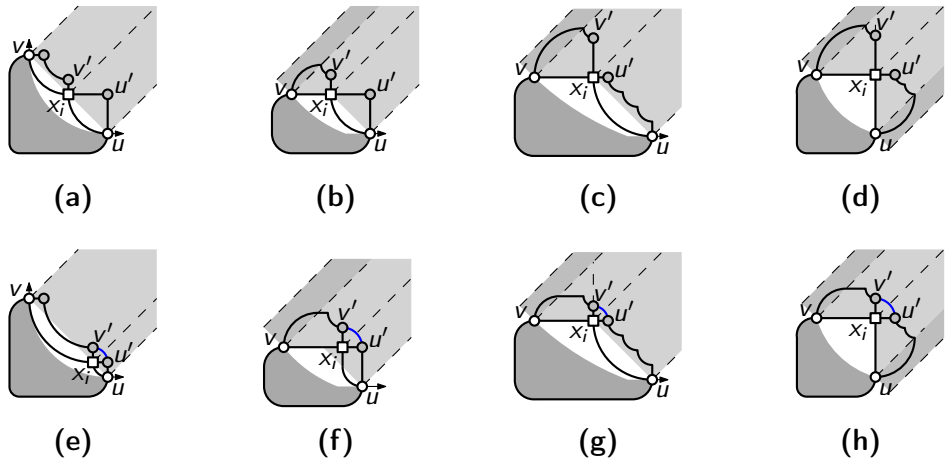


Figure 4.16: Drawing of a facial pair of an in-dummy x_i. (a)–(d) Possibly occuring cases for the drawings of f_2 and f_3. (e)–(h) Due to changes of edge lengths, reference edge (u', v') is realized as a quarter circular arc.

We finish the drawing of the facial pair in two steps: First, we choose a suitable drawing for both f_2 and f_3 from the potential drawings of normal faces; see Fig. 4.14; this choice is determined based on the shapes of reference edges (x_i, u) and (x_i, v) and the number of vertices on f_2 and f_3; see Fig. 4.16a to 4.16d for some subcases.

Note that each of faces f_2 and f_3 can be either triangular or of larger length which determines whether we use one of the drawings in Figs. 4.14a to 4.14d (or 4.14i to 4.14j if there are virtual side-arcs) or 4.14e to 4.14h (or 4.14k to 4.14o if there are virtual side-arcs), respectively.

Finally, it remains to add edge (u', v'). Note that v' and u' are part of faces f_2 and f_3, respectively. We align v' and u' on a common diagonal and realize (u', v') as a convex quarter circular arc as follows:

- If edges (u, x_i) and (v, x_i) are both concave quarter arcs, we move x_i along the diagonal through u and v; 4.16a. When moving x_i towards u, the length of the vertical segment (x_i, v') increases w.r.t. to the length of the horizontal segment (x_i, u'). Hence, we find a placement where they have the same length; see Fig. 4.16e.
- If edge (v, x_i) is a horizontal segment and edge (x_i, u) is a quarter circular arc, we distinguish two cases. If the length of (x_i, v') is smaller than the length of (x_i, u'), we move x_i horizontally towards u'; see Fig. 4.16b. As a result, we redraw (u, x_i) with two segments, with a quarter circular arc incident to u and a vertical segment incident to x_i; see Fig. 4.16f. Otherwise, the length of (x_i, v') is larger than the length of (x_i, u'); see Fig. 4.16c. Then, we redraw the side-arc incident to v with two segments such that a

quarter circular arc is incident to v and is followed by a horizontal segment; see Fig. 4.16g.

– The case where edge (v, x_i) is a quarter-circular arc and (x_i, u) is a vertical segment is symmetrical to the previous case.

– If (u, x_i) and (v, x_i) are both straight-line segment, we shorten the longer segment from (x_i, v') and (x_i, u'); see Fig. 4.16d. In the process, we redraw the corresponding side-arc as in the previous case; see Fig. 4.16h.

Note that it may be required to redraw one of reference edges (x_i, u) and (x_i, v) as a sequence of a quarter-circular arc and a straight-line segment incident to x_i; see Fig. 4.16f. Since however half-edges (x_i, u') and (x_i, v') are always realized as a straight-line segment, the two intersecting edges (u, v') and (v, u') of G are drawn with at most two segments, which satisfies Invariant 6 for them.

Since x_i is a dummy, neither u' nor v' can be a dummy and Invariant 5 is fulfilled. If (u', v') is a virtual reference edge for f_4, it fulfills Invariant 2; otherwise it fulfills Invariant 6 for f_4. If a side-arc is drawn with two segments it fulfills Invariant 1 while Invariant 6 still holds for faces f_2 and f_3. All remaining edges are drawn according to one of the cases discussed in the previous case, hence, they do not violate an invariant.

(iii) f **is cut-face.** Recall that f consists of 4-cycle (u, v, v', u'); see Fig. 4.12a. Instead of drawing f, we immediately draw the two intersecting edges (u, v') and (v, u'). If reference edge (u, v) is not virtual, it is drawn either as a horizontal segment or as a quarter circular arc by Invariant 3. Moreover, it cannot be a side-arc as in such a case, one of u or v would have no free port left. Here, we use one of the drawings from Figs. 4.17a to 4.17d for edges (u, v') and (v, u'). If edge (u, v) is a virtual reference edge and no side-arc, we instead use one of the drawing from Figs. 4.17e to 4.17h. Finally, if (u, v) is a virtual side-arc, we use one of the two drawings from Figs. 4.17i and 4.17j; which can always be done by Invariant 4. Note that in either case, u' and v' are diagonally aligned so that (u', v') can be realized as a concave quarter circle. This guarantees Invariants 2 and 3. Since no side-arcs and convex arcs are introduced, Invariants 1, 4 and 5 hold. Finally, edges (u, v') and (v, u') satisfy Invariant 6.

It remains to discuss that the output is indeed an embedding preserving SC_2-drawing. We start by proving that all edges are drawn with at most two segments. By Invariant 6 the property holds for all edges except those that cross at out-dummies. So consider a pair of edges (u, v'), (v, u') that intersects at an out-dummy x_o. By Invariant 1, a half-edge h incident to x_o has two segments only if it is a side-arc incident to the south or west port of x_o. Moreover, we only encounter this scenario when drawing a facial pair; see Fig. 4.16. Assume w.l.o.g. that h

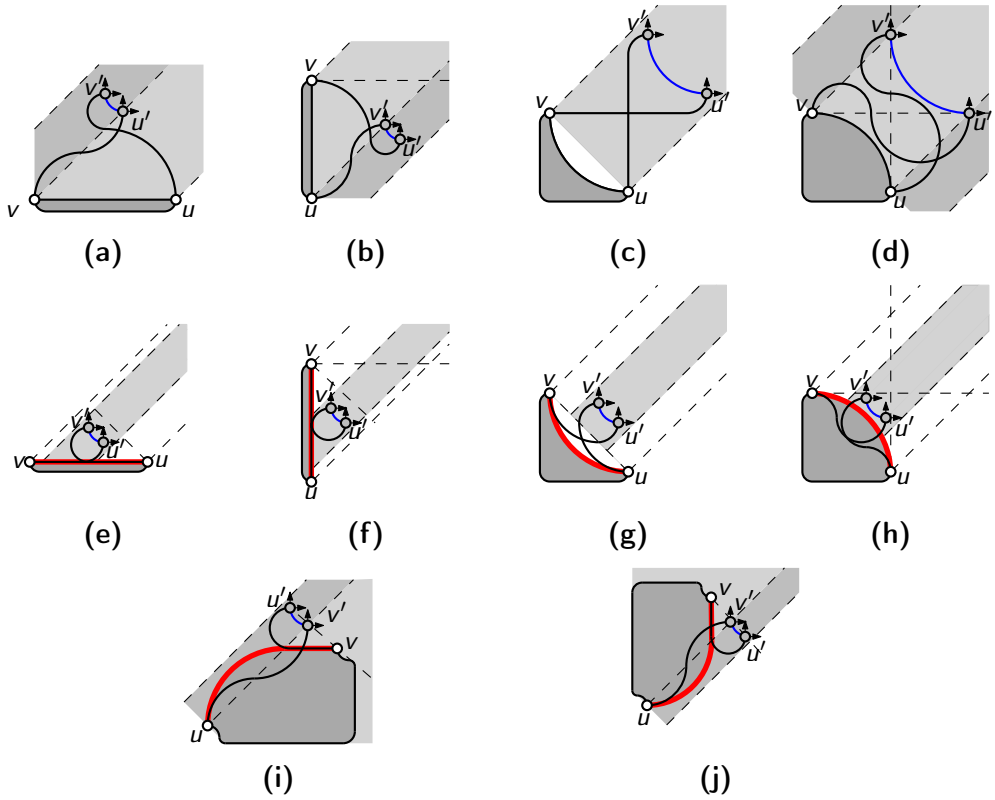


Figure 4.17: Drawing of the pair of intersecting edges (u, v'), (v, u') when encountering a cut-face. (a)–(d) Cases where reference edge (u, v) is part of G. (e)–(h) Cases where reference edge (u, v) is a virtual reference edge but not a side-arc. (i)–(j) Cases where reference edge (u, v) is a virtual side-arc.

is incident to the west port of x_o. Then h is attached to x_o with a horizontal segment. Consider the second half-edge h' that is connected to the east port of x_o and forms an edge with h. By Invariant 1, it is drawn as a single segment which is not a convex quarter-circular arc by Invariant 5. So, assume it was a side-arc. Then, both h and h' are on the outer face. This however contradicts the fact, that for an out-dummy exactly two consecutive incident half-edges are on the outer face. Hence, h' is a horizontal segment and e is composed of two segments.

We ensure the 1-planar embeddings with Invariants 7 and 8. By Invariant 8, we ensure that we can always insert new faces on the current outer face. Moreover, if all arcs of a face inserted at the same time are chosen with the same radius, Invariant 7 ensures that the new subgraphs inserted at such reference edges do not overlap due to the corresponding widths of the different semi-strips. We remark that the widths of semi-strips for child components decreases exponentially, hence, we cannot claim polynomial area for the produced drawings. □

Finally, we show that the curve complexity bounds achieved by our results in Theorem 4.4 and 4.5 are worst-case optimal.

Theorem 4.6. *For every $k \geq 2$, there exists a biconnected outer-1-planar graph G_k with maximum degree four and fixed embedding $\mathcal{E}_k$ on $4k$ vertices that admits no 1-planar OC_2-drawing realizing $\mathcal{E}_k$. In addition, $\Omega(k)$ edges of complexity at least three are required in any embedding preserving OC_3-drawing.*

Proof. Consider K_4 with an outer-1-planar embedding. As every vertex must have at least one free port to the interior of the outer cycle, there are at least four bends on the outer face. Therefore, each edge on the outer face has one bend in an embedding preserving OC_2-drawing. As a result, the remaining two edges must be drawn without a bend since they must have an even number of bends; see Fig. 4.18a. We conclude that K_4 has a unique outer-1-planar OC_2-drawing.

Next, consider two copies G_1 and G_2 of K_4 and let e_1 be an edge on the outer face of G_1 and e_2 be an edge on the outer face of G_2. We can create a biconnected outer-1-planar graph G by connecting the endpoints of e_1 with one of the endpoints of e_2 each such that the two resulting edges e_1' and e_2' intersect; see Fig. 4.18b. Since e_1 and e_2 must be drawn with a bend per edge each in an outer-1-planar drawing of G, we can only draw one of e_1' and e_2' with at most one bend per edge; see Fig. 4.18b.

Clearly, we can repeat the previous construction by connecting k copies of K_4 in a chain as shown in Fig. 4.18c. By the previous argument, for each independent pair of two consecutive copies of K_4 at least one edge must be drawn with at least two bends, which proves the theorem. □

Theorem 4.7. *For every $k \geq 2$, there exists a biconnected outer-1-planar graph G_k with maximum degree four and fixed embedding $\mathcal{E}_k$ on $4k + 2$ vertices that*

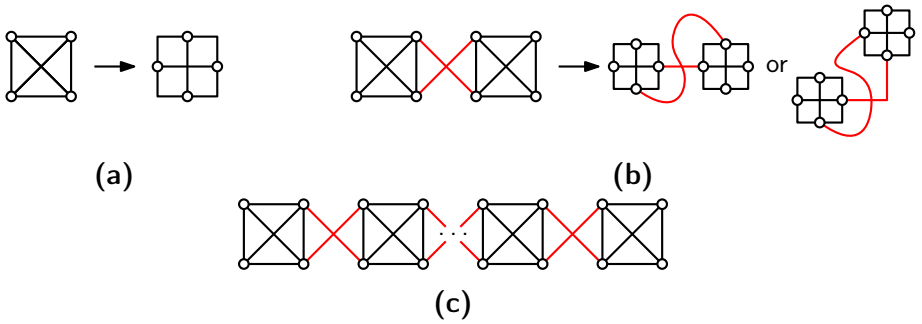


Figure 4.18: (a) K_4 has a unique outer-1-planar OC_2-drawing. (b) Connecting two K_4's with a pair of intersecting edges creates an outer-1-planar graph that has no OC_2-drawing with fixed embedding. (c) A construction that can be repeated to create an infinite family of graphs that requires $\Omega(n)$ edges with at least three segments.

Figure 4.19: (a) An outer-1-planar drawing of K_4 is composed of four planar triangles that share one edge each with every other triangle. If the free ports of K_4 are required to be located on the exterior, such a triangle must have two vertices whose two free ports are outside the triangle and one vertex which has one free port outside and inside the triangle each. (b)–(d) All valid SC_1-drawings of a triangle fulfilling the properties.

admits no 1-planar SC_1-drawing realizing $\mathcal{E}_k$. In addition, $\Omega(k)$ edges of complexity at least two are required in any embedding preserving SC_2-drawing.

Proof. Our counterexample follows closely the construction from the proof of Theorem 4.6. Again, we first consider an outer-1-planar embedding of K_4. In order to restrict how this embedding can be realized with an SC_1-drawing, we additionally require that all its free ports are located on the outer face. We now observe that such a drawing of K_4 is composed of four planar triangles which share one edge each with each other triangle; see Fig. 4.19a. Let T be such a triangle. Since the K_4 containing T has all free ports on its exterior, two of the vertices of T must have all free ports outside T, while the remaining vertex has one free port inside T and outside T each.

Since one free port will be located inside T, the sum of angles at vertices inside T is equal to 2π. However, the sum of interior angles of a triangle must be equal to π. The difference of π must be accommodated by a convex curvature of edges incident to T. Table 4.2 lists all triples of edge curvatures and whether those curvatures can be combined into an SC_1-drawing for T. Since every edge of T has

Table 4.2: Overview of all triples of edge curvatures for a triangle T that yield a total curvature of π and information whether such a triple admits an SC_1-drawing of T. Positive curvatures are convex, negative curvatures are concave.

Edge Curvatures	Realizable?
$\frac{\pi}{2}$, $\frac{\pi}{2}$, 0	yes, see Fig. 4.19b
π, 0, 0	yes, see Fig. 4.19c
π, $\frac{\pi}{2}$, $-\frac{\pi}{2}$	yes, see Fig. 4.19d
π, π, $-\pi$	no
$\frac{3\pi}{2}$, 0, $-\frac{\pi}{2}$	no
$\frac{3\pi}{2}$, $\frac{\pi}{2}$, $-\pi$	no
$\frac{3\pi}{2}$, π, $-\frac{3\pi}{2}$	no

to be shared with an edge of another triangle T', we observe that the drawings in Figs. 4.19c and 4.19d cannot be part of an SC_1-drawing of K_4. In particular, if T and T' share the half circular arc, the two remaining vertices overlap if both T and T' use the same drawing. If T and T' use the drawings of Fig. 4.19c and of Fig. 4.19d, respectively, the third vertex of T will be located entirely inside T', which contradicts outer-1-planarity. Hence, all triangles use the drawing of Fig. 4.19b. It follows that there is a unique outer-1-planar SC_1-drawing of K_4 with all free ports on the outer face, see Fig. 4.18a.

As in the proof of Theorem 4.6, we connect k copies of K_4 in a chain via pairs of intersecting edges as shown in Fig. 4.20b. In order to force the ports of each copy to be located on the outer face, we introduce two more degree two vertices at the ends of the chain. The resulting graph is still biconnected. Again, we will show that each independent pair of consecutive copies G_1 and G_2 of K_4 requires at least one edge that must be drawn with at least one bend in any SC_2-drawing.

Hence, consider such a pair of copies G_1 and G_2 of K_4. Let e_1 be an edge on the outer face of G_1 and e_2 be an edge on the outer face of G_2. Recall that G_1 and G_2 must be drawn such that all free ports are on the outer face. The endpoints of e_1 are connected with one of the endpoints of e_2 each such that the two resulting edges e_1' and e_2' intersect; see Fig. 4.20c. Since e_1 and e_2 must be drawn as a quarter circular arc in any outer-1-planar SC_1-drawing of G_k, we can only draw one of e_1' and e_2' with exactly one segment; see Fig. 4.20c. Note that vertices located on dashed lines in Fig. 4.20c must be necessarily aligned if e_1' is drawn with a single segment. This immediately results in e_2' being not drawable with a single segment. This completes the proof. □

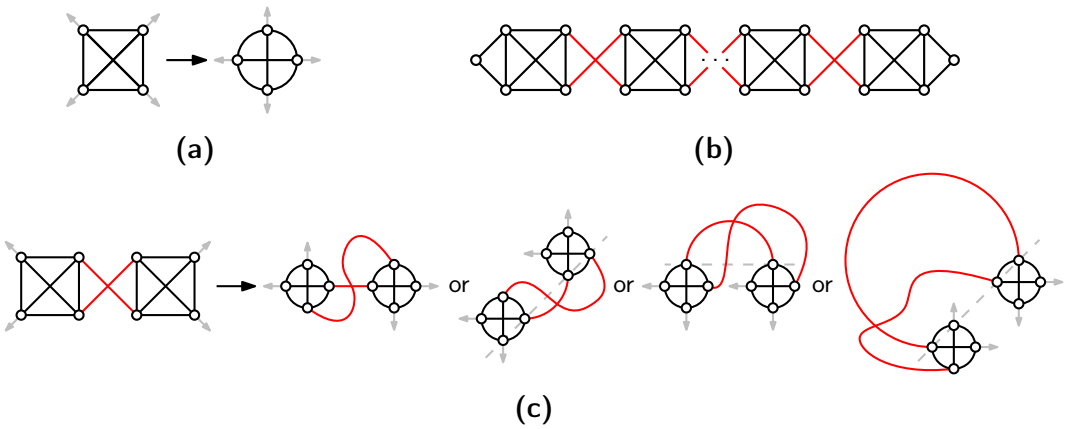


Figure 4.20: (a) K_4 has a unique outer-1-planar SC_1-drawing where all free ports are on the outer face. (b) The construction from Fig. 4.18c can be adjusted to require the free ports of all K_4's to be on the outer face by introducing two more degree two vertices at the end of the chain. (c) Connecting two K_4's with all free ports on the outer face with a pair of intersecting edges creates an outer-1-planar graph that has no SC_1-drawing with fixed embedding. The gray dashed lines indicate vertices that are required to be placed on the same line.

Part II

Beyond Stack Layouts

Queue Layouts and Arc Diagrams of Planar Graphs

Chapter 5

Queue Layouts of Bounded Degree Planar Graphs

A linear layout of a graph consists of a total ordering $\prec$ of the vertices and an assignment of the edges to k different color classes called *pages*. In a *k-page queue layout* no two edges (u, v) and (u', v') of the same color class nest each other, that is, $u \prec u' \prec v' \prec v$ is a forbidden pattern. Queue layouts have diverse applications such as scheduling [42], sorting [153], VLSI design [126], and graph drawing [68, 78]. An important parameter in such applications is the *queue number* $qn(G)$ of a graph G, that is, the minimum integer value such that G admits a k-page queue layout. While for the complete graph $qn(K_n) = \Theta(n)$ [108], in many situations better bounds for the queue number are possible: First, graphs with m edges have queue number $\mathcal{O}(\sqrt{m})$ [107]. Second, any minor-closed graph family has polylogarithmic queue number [78]. Finally, the queue number is also bounded by tree- and pathwidth [78, 158], tracknumber [81], bandwidth [107] and layered pathwidth [23].

For the class of planar graphs on the other hand, mostly restricted subclasses have been investigated in the past. Namely, it has been shown that the graphs admitting 1-page queue layouts are indeed a subclass of planar graphs [108] while also outerplanar graphs [107], series parallel graphs [143] and Halin graphs [98] have bounded queue number. The best known upper bound of $\mathcal{O}(\log n)$ for planar graphs on the other hand has been derived from the layered pathwidth of planar graphs [23] – this result is obtained without explicitly using the planarity of the graph. Recently, the upper bound on the queue number of planar 3-trees has been improved [8] using arguments that build upon planarity.

In this chapter[1], we provide a strong evidence for the correctness of the conjecture by Heath, Leighton and Rosenberg that the class of planar graphs has a bounded queue number [107] which was open until recently.[2] Namely, we show

[1] The results of this chapter also appeared in [25].

[2] We acknowledge that more recently the conjecture has been shown to be correct [76].

that every planar graph of maximum degree Δ admits a $\mathcal{O}(\Delta^c)$-page queue layout for a small constant value of c. In particular, our result relies on the input graph being planar and stands in contrast to the fact that, in general, graphs of bounded maximum degree have arbitrarily large queue number [160]. More precisely, we show:

Theorem 5.1. *Let G be a planar graph of maximum degree Δ. Then, it holds that $qn(G) \leq 32(2\Delta - 1)^6 - 1$.*

Moreover, our proof is constructive and indicates that such a queue layout can be computed in polynomial time as discussed in Section 5.4. Before we describe our result, we first discuss some notable implications of Theorem 5.1.[3]

First, consider the so-called *track layouts*. In a *k-track layout*, vertices are partitioned into k subsets called *tracks*. The vertices on track t are linearly ordered with a total ordering $\prec_t$ such that the edges between each pair of tracks do not intersect, that is, for two edges (u, v) and (u', v') with u and u' on track t_1 and v and v' on track t_2, it holds that $u \prec_{t_1} u'$ if and only if $v \prec_{t_2} v'$. The *track number* $tn(G)$ of a graph G is equal to the minimum number of tracks such that G has a $tn(G)$-track layout. The best upper bound for the track number of planar graphs is $\mathcal{O}(\log n)$ [23].[3] It is known that a graph with queue number q and acyclic chromatic number k has track number at most $k(2q)^{k-1}$ [78]. Moreover, planar graphs have acyclic chromatic number at most five [46]. Together with Theorem 5.1 we obtain:

Corollary 5.1. *Let G be a planar graph of maximum degree Δ. Then, it holds that $tn(G) = \mathcal{O}(\Delta^{24})$.*

Second, consider three-dimensional straight-line grid drawings. Every c-vertex colorable graph with n vertices and track number t admits a three-dimensional straight-line grid drawing in volume $\mathcal{O}(c) \times \mathcal{O}(ct) \times \mathcal{O}(c^5 n)$ [80]. A similar reverse relation is also known, namely, if G has a three-dimensional straight-line grid drawing in volume $X \times Y \times Z$, it holds that $tn(G) \leq 2XY$ [77]. The question whether a planar graph has a three-dimensional straight-line grid drawing in linear volume is a major open problem in graph drawing and has been asked in 2003 [91]. The best known upper bound for planar graphs is $\mathcal{O}(n \log n)$ volume [23][3], which we can improve in the restricted degree case:

Corollary 5.2. *Let G be a planar graph of maximum degree Δ. Then, G admits a three-dimensional straight-line grid drawing in volume $\mathcal{O}(1) \times \mathcal{O}(\Delta^{24}) \times \mathcal{O}(n)$.*

Third, consider the *2-track thickness* of a bipartite graph G. Here the problem is to draw the vertices of each partition of G on one of two parallel lines and

[3] Note that the implications discussed also can be improved and generalized to all planar graphs using the result from [76].

(a) (b)

Figure 5.1: Different patterns formed by edges in a linear ordering: (a) 3-necklace, and, (b) 3-rainbow.

then color the edges of G such that no two edges of the same color intersect in a straight-line drawing. The minimum number of colors that suffices is called the *2-track thickness* $\theta_2(G)$. It is known that $\theta_2(G)$ is directly related to $qn(G)$ [81] which implies the best known upper bound of $\mathcal{O}(\log n)$ for planar graphs [23].[3] For the case of bounded degree we obtain the following:

Corollary 5.3. *Let G be a planar graph of maximum degree Δ. Then, it holds that $\theta_2(G) = \mathcal{O}(\Delta^6)$.*

Finally, we point out that Theorem 5.1 can be generalized for k-planar graphs where k is a constant according to a result in [81]:

Corollary 5.4. *Let G be a k-planar graph of maximum degree Δ. Then, it holds that $qn(G) = \Delta^{\mathcal{O}(k)}$.*

The remainder of the chapter is organized as follows: First, in Section 5.1, we introduce some additional helpful lemmas and concepts from the literature. Then, in Section 5.2, we show that Δ-*matched graphs* which are a subfamily of planar graphs of maximum degree Δ have queue number at most $2\Delta - 2$. Based on this result, we generalize to all planar graphs of maximum degree Δ in Section 5.3 before discussing the time complexity of our layout algorithm in Section 5.4.

5.1 Tools froms the Literature

Results on Queue Layouts. Consider a linear ordering $\prec$ of the vertices of a graph G and k edges $(u_1, v_1), \ldots, (u_k, v_k)$ that pairwise do not share any endpoint. If $(u_1, v_1), \ldots, (u_k, v_k)$ are pairwise independent, that is, w.l.o.g. $u_1 \prec v_1 \prec u_2 \prec v_2 \prec \ldots \prec u_k \prec v_k$, we say they form a *$k$-necklace* in $\prec$; see Fig. 5.1a. If each pair of edges from $(u_1, v_1), \ldots, (u_k, v_k)$ is such that one edge nests the other, that is, w.l.o.g. $u_1 \prec \ldots \prec u_k \prec v_k \prec \ldots \prec v_1$, we say that edges $(u_1, v_1), \ldots, (u_k, v_k)$ form a *k-rainbow* in $\prec$; see Fig. 5.1b. It is known that a graph G admits a k-page queue layout Γ if and only if it admits a linear order $\prec$ of its vertices in which no $(k+1)$-rainbow occurs [108]; more precisely, such a linear order $\prec$ will be used by Γ. On the other hand, a k-necklace that occurs in the linear order of a queue layout can be realized on the same page of the queue layout for any k.

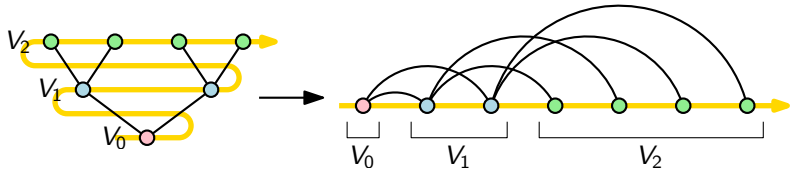


Figure 5.2: Computation of a queue layout of a tree.

Next, we describe how to compute a 1-page queue layout of a tree T according to [108][4]. Assume w.l.o.g. that r is the root of T and let h denote the height of T, i.e., the maximum graph-theoretic distance between r and any other vertex in T. Further, let $V_0, \ldots, V_h$ be a partition of the vertices such that $v \in V_i$ if the graph-theoretic distance between v and r is equal to i. In addition, let $\prec_0, \ldots, \prec_h$ be linear orderings of the vertices in $V_0, \ldots, V_h$, respectively, such that for any pair of edges (u, v) and (u', v') with $u, u' \in V_i$ and $v, v' \in V_{i+1}$ it holds that $u \prec_i u'$ if and only if $v \prec_{i+1} v'$. Then we create a linear ordering $\prec$ of all vertices V by merging orderings $\prec_0, \ldots, \prec_h$ such that $v \prec v'$ if $v \in V_i$ and $v' \in V_j$ for $i < j$; see Fig. 5.2. We call such a linear ordering $\prec$ a *good tree ordering*. More precisely, we can define a good tree ordering as follows.

Definition 5.1. *Let $F = (V, E)$ be a forest and $V_0, \ldots, V_h$ be a partititoning of V into independent sets such that $(u, v) \in E$ if and only if w.l.o.g. $u \in V_i$ and $v \in V_{i+1}$ for some $0 \leq i \leq h$. A linear ordering $\prec$ of V is a good tree ordering if the following holds:*

T.1 Let $u \in V_i$ and $v \in V_j$ be vertices such that $i < j$. Then $u \prec v$.

T.2 Let $(u, v), (u', v') \in E$ such that $u, u' \in V_i$ and $v, v' \in V_{i+1}$. Then $u \prec u'$ if and only if $v \prec v'$.

Consider two edges (u, v) and (u', v') such that $u \preceq u'$, $u \prec v$ and $u' \prec v'$. If $u \in V_i$ and $u' \in V_j$ for $j > i$, it follows that $v \prec v'$ due to Property T.1. Hence, (u, v) cannot nest (u', v'). Otherwise, consider the case where $u, u' \in V_i$ and $v, v' \in V_{i+1}$. Here, we know from Property T.2 that $v \prec_{i+1} v'$ since $u \preceq_i u'$. Again, it is not possible that (u, v) nests (u', v''). We conclude that $\prec$ contains no 2-rainbow implying the following:

Lemma 5.1 ([108]). *Let $G = (V, E)$ be a forest and $\prec$ a good tree ordering of V. Then G admits a 1-page queue layout with linear ordering $\prec$.*

Finally, we point out that queue layouts are known to have a nice behavior regarding *subdivisions*. Consider two graphs $G = (V, E)$ and $G' = (V', E')$. We say that G' is obtained from G by *subdividing edge $e = (u, v) \in E$ k times* if and

[4] In [108], the result is actually shown for the class of *arched-level planar graphs* which are a superclass of trees.

Figure 5.3: (a) The octahedral graph with a BFS-tree (bold) rooted a vertex 1 (red). (b) The corresponding ordered concentric representation.

only if G' consists of $G \setminus e$ and a path p_e of length $k+1$ between u and v such that u and v are the only vertices along p_e that are part of G. In a *k-subdivision* G' of a graph G, every edge e of G is subdivided at most k times. It is known, that a graph has bounded queue number if one of its subdivisions has:

Lemma 5.2 ([81]). *Let G' be a k-subdivision of a graph G and let $qn(G') = q$. Then $qn(G) \leq \frac{1}{2}(2q+2)^{2k} - 1$.*

Ordered Concentric Representations. *Ordered concentric representations* have been recently introduced in the study of linear layouts [139]. Let G a planar graph and r a vertex of G. In an ordered concentric representation Γ of a planar graph G with *center* r, vertices are restricted to concentric circles $\mathcal{C}_0, \ldots, \mathcal{C}_h$ where h is the maximum smallest graph-theoretic distance between r and any vertex in G such that the following properties hold:

O.1 Vertex v is located on $\mathcal{C}_i$ if and only if $dist(r, v) = h - i$. In particular, $\mathcal{C}_h$ contains only r and can be considered as a single point. As a consequence, each edge e of G is either connecting two vertices on the same circle or two vertices on consecutive circles. In the former case, we call e a *level edge*; in the latter case, we call it a *binding edge*.

O.2 Vertex r is incident to the outer face.

O.3 Each level edge is drawn outside the circle to which its two endpoints are restricted to.

O.4 Each binding edge has a segment between the two circles to which its two endpoints are restricted to and potentially a second segment that is drawn outside those two circles.

Ordered concentric representation are computed by first computing a BFS-tree rooted at a vertex r on the outer face. Root r is then placed on $\mathcal{C}_h$, while every other vertex v is placed on $\mathcal{C}_i$ where $i = h - dist(r, v)$. The embedding of the

input graph can be maintained. In particular, the edges of the BFS-tree become binding edges which are drawn entirely between both circles; cf. Property O.4. More formally, we can summarize this as follows:

Lemma 5.3 ([139]). *Let G be a planar graph on n vertices and r one of its vertices. An ordered concentric representation with center r where vertices are restricted to circles $\mathcal{C}_0, \ldots, \mathcal{C}_h$ can be computed in $\mathcal{O}(n)$ time where h is the height of a BFS-tree of G rooted at r.*

Since we will make use of ordered concentric representations throughout this chapter, we introduce some shortened notation. Consider a BFS-tree T of a graph G with height h rooted at a vertex r. For vertex v, we denote by $dist(v)$ the graph-theoretic distance between v and r in T. Note that $dist(r) = 0$ while there exists a leaf l with $dist(l) = h$. Moreover, we call the value $h - dist(v)$ the *layer* $\ell(v)$. In particular, $\ell(r) = h$ while there exists a leaf l with $\ell(l) = 0$. Also note that in an ordered concentric representation, vertex v is located on $\mathcal{C}_i$ if and only if $\ell(v) = i$.

5.2 Δ-Matched Graphs

We first consider a special subclass of planar graphs of bounded degree Δ, called *Δ-matched graphs*.

Definition 5.2. *A Δ-matched graph $G = (V, E)$ consists of a complete $(\Delta - 1)$-ary spanning tree T with root $r \in V$ and a perfect matching M of its leaves such that:*

P.1 All leaves have the same graph-theoretic distance to r.

P.2 G admits an ordered-concentric representation with BFS-tree T and r on the innermost circle such that all edges of T are drawn with a single segment (cf. Property O.4) while all edges of M are drawn outside of $\mathcal{C}_0$.

Refer to Fig. 5.4 for an illustration of a 3-matched graph. Note that a Δ-matched graph extends a tree in such a way that a good tree ordering $\prec$ for T may introduce k-rainbows for arbitrary integers k in the perfect matching M. In the following, we will define a total ordering $\prec$ of the vertices and a $(2\Delta - 3)$-coloring of the edges of T such that $\prec$ is a good tree ordering for each of the forests induced by the set of edges of one color while M forms a k-necklace in $\prec$ for some integer k. Then, using $\prec$ and Lemma 5.1, we can draw each of the $(2\Delta - 3)$ subforests in one page each and matching M in a separate page achieving a $(2\Delta - 2)$-page queue layout of G. We point out that this result cannot be easily obtained by considering the treewidth of G as the treewidth of Δ-matched graphs

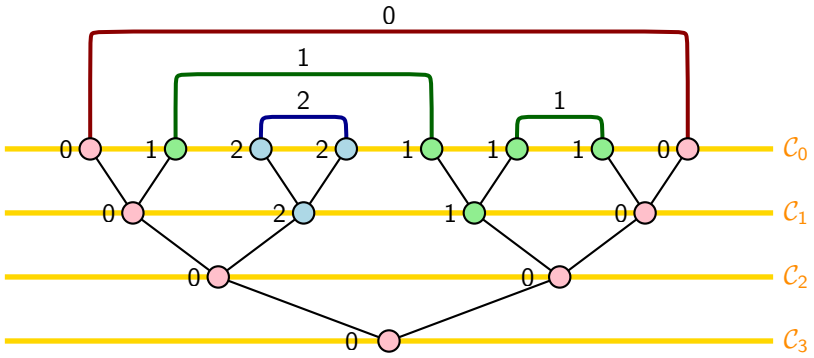


Figure 5.4: A 3-matched graph G in an ordered concentric representation according to Property P.2. Matching edges are labeled with their nesting value while vertices are labeled with their matching value. In this example every inner vertex has degree exactly Δ, which is not necessary in general.

is not bounded by a constant. In particular, a suitable subdivision of a grid graph gives a 4-matched graph and has treewidth $\Theta(\sqrt{n})$.

First, we define a linear order for the vertices on each of the circles $\mathcal{C}_0, \ldots, \mathcal{C}_h$ of the ordered concentric representation of G where h is the height of tree T. Since r is part of the outer face, there exists a curve c starting at r and intersecting each of circles $\mathcal{C}_0, \ldots, \mathcal{C}_{h-1}$ exactly once. For $\mathcal{C}_i$, we define the linear order $\prec_i$ of the vertices on $\mathcal{C}_i$ according to the order in which we encounter them in a clockwise traversal of $\mathcal{C}_i$ starting from the intersection between $\mathcal{C}_i$ and c. In particular, since each edge of T is drawn with a single segment and since the ordered-concentric representation is planar, we have the following additional property:

P.3 Let (u, v) and (u', v') be two edges of the tree such that u and u' appear on $\mathcal{C}_i$ and v and v' appear on $\mathcal{C}_{i-1}$. Then, $u \prec_i u'$ if and only if $v \prec_{i-1} v'$.

Observe that this is basically Property T.2 of a good tree ordering.

Next, we consider the edges belonging to M and assign an integer value to each of them that we call the *nesting value*. For edge $e \in M$, the nesting value $nv(e)$ is equal to the number of edges that nest e in $\prec_0$. Afterwards, we assign an integer value to each vertex $v \in V$ which we call the *matching value* $mv(v)$. We assign the matching values in a bottom-up traversal of the tree T from $\mathcal{C}_0$ to $\mathcal{C}_h$. First, for a vertex v on $\mathcal{C}_0$, we set $mv(v) = nv(e)$ where e is the unique edge in M incident to v. Then, for a vertex v on $\mathcal{C}_i$, the matching value $mv(v)$ is equal to the minimum matching value of any of its neighbors on $\mathcal{C}_i$. The nesting and matching values are labeled in the example in Fig. 5.4. Intuitively speaking, the matching value of vertex v is equal to the minimum nesting value of an edge in M that is incident to a leaf in the subtree of T rooted at v.

We point out that the matching values of leaves which are consecutive in $\prec_0$ differ by at most one. Therefore, if the subtree of T rooted at v has two leaves

with matching values α and β with $\alpha < \beta$, then it also has at least one leaf with matching value m for every $m \in [\alpha, \beta]$. Since a vertex v on layer $\ell(v)$ and hence on circle $\mathcal{C}_{\ell(v)}$ can be incident to at most $(\Delta-1)^{\ell(v)}$ leaves, the matching values of its leaves differ by at most $(\Delta-1)^{\ell(v)}-1$. As a result, two vertices u and v on layer $\ell(v)$ behave differently, if their matching values differ by $(\Delta-1)^{\ell(v)}$. We capture this behavior by assigning each vertex v to a *layer group* $g(v)$ such that

$$g(v) = \left\lfloor \frac{mv(v)}{(\Delta-1)^{\ell(v)}} \right\rfloor. \tag{5.1}$$

In particular, for vertices in layer 0, it holds that $mv(v) = g(v)$, while in general $mv(v) \in \left[g(v)(\Delta-1)^{\ell(v)}, (g(v)+1)(\Delta-1)^{\ell(v)}\right)$. We denote by V_ℓ^g the vertices of G for which the layer is ℓ and the layer group is g. It is easy to see that $\{V_\ell^g\}_{\ell,g}$ partitions the vertices of G. We are now ready to prove the following lemma:

Lemma 5.4. *Let G be a Δ-matched graph. Then $qn(G) \leq 2\Delta-2$.*

Proof. Let $\{V_\ell^g\}_{\ell,g}$ be a partition of the vertices and let $\prec_i$ be a linear ordering of the vertices of layer i as described above. We construct the linear ordering $\prec$ of the vertices such that $u \prec v$ if and only if one of the following conditions holds:

C.1 $\ell(u) < \ell(v)$, or,

C.2 $\ell(u) = \ell(v)$ and $g(u) < g(v)$, or,

C.3 $\ell(u) = \ell(v) =: \ell$ and $g(u) = g(v)$ and $u \prec_\ell v$.

We first show that M forms a necklace in $\prec$. To see this, consider an edge $e \in M$. Since the matching value of vertices in layer 0 is equal to the corresponding nesting value of the incident matching edge, both endpoints of e belong to the same layer group. In addition, e only nests edges with larger nesting values. Since vertices of the same layer group are ordered according to $\prec_0$ by Condition C.3, the endpoints of e are indeed consecutive in $\prec$. Hence, we can assign all edges of M to the same page in a queue layout with ordering $\prec$.

It remains to assign edges belonging to T to some page. In order to do so, we show that we can partition the edges of T to $2\Delta-3$ disjoint forests (which each can be augmented to a tree) for which $\prec$ is a good tree ordering and use one page for each of those forests; see Lemma 5.1. We first consider a vertex u belonging to layer $\ell+1$ and establish to how many different layer groups of layer ℓ it can be incident to. So let $u \in V_{\ell+1}^g$ and let $v_1, v_2, \ldots, v_d$ be the neighbors of u in layer ℓ where $d \leq \Delta$. Assume without loss of generality that $mv(v_1) \leq mv(v_2) \leq \ldots \leq mv(v_d)$, i.e., it is not necessarily true that $v_1 \prec_\ell v_2 \prec_\ell \ldots \prec_\ell v_d$. Clearly, $mv(u) = \min\{mv(v_i) | i = 1, \ldots, d\} = mv(v_1)$. By $u \in V_{\ell+1}^g$ and Eq. (5.1),

$$mv(v_1) = mv(v) \in \left[g \cdot (\Delta-1)^{\ell+1}, (g+1) \cdot (\Delta-1)^{\ell+1}\right).$$

In other words,

$$\begin{aligned} g \cdot (\Delta-1)(\Delta-1)^{\ell} &= g \cdot (\Delta-1)^{\ell+1} \\ &\leq mv(v_1) \\ &< (g+1) \cdot (\Delta-1)^{\ell+1} \\ &= (g+1) \cdot (\Delta-1)(\Delta-1)^{\ell}, \end{aligned} \tag{5.2}$$

that is,

$$g \cdot (\Delta-1) \leq g(v_1) < (g+1) \cdot (\Delta-1).$$

Alternatively, we can write

$$v_1 \in \bigcup_{k=0}^{\Delta-2} V_{\ell}^{g\cdot(\Delta-1)+k}.$$

Next, we show that for vertex v_i with $1 \leq i \leq d$, it holds that

$$g \cdot (\Delta-1) \leq g(v_i) < (g+1) \cdot (\Delta-1) + (i-1). \tag{5.3}$$

Recall first that the matching values of two consecutive leaves differ by at most one. Therefore, $mv(v_i)$ and $mv(v_{i-1})$ differ at most by the number of leaves in the subtree rooted at v_{i-1}. Since this subtree has at most $(\Delta-1)^{\ell}$ leaves, it follows that

$$mv(v_i) \leq mv(v_{i-1}) + (\Delta-1)^{\ell}. \tag{5.4}$$

Applying Eq. (5.4) recursively, we obtain

$$mv(v_i) \leq mv(v_1) + (i-1)(\Delta-1)^{\ell},$$

which by Eq. (5.2) yields

$$\begin{aligned} mv(v_i) &< (g+1) \cdot (\Delta-1)(\Delta-1)^{\ell} + (i-1)(\Delta-1)^{\ell} \\ &= ((g+1) \cdot (\Delta-1) + (i-1))(\Delta-1)^{\ell}. \end{aligned}$$

Because $mv(v_1) \leq mv(v_i)$ for $i > 1$ we obtain

$$g \cdot (\Delta-1)(\Delta-1)^{\ell} \leq mv(v_i) < ((g+1) \cdot (\Delta-1) + (i-1))(\Delta-1)^{\ell},$$

which is equivalent to our claimed Eq. 5.3. Since $i \leq d \leq \Delta-1$, we obtain

$$v_i \in \bigcup_{k=0}^{2\Delta-4} V_{\ell}^{g\cdot(\Delta-1)+k}. \tag{5.5}$$

We assign each edge (u, v) of T to one of the forests $F_1, \ldots, F_{2\Delta-3}$ as follows: Let $u \in V_{\ell+1}^{g}$ and $v \in V_{\ell}^{g\cdot(\Delta-1)+k}$. Then, we assign edge (u, v) to forest F_{k+1}. Note that by Eq. (5.5), we can assign each edge to one of forests $F_1, \ldots, F_{2\Delta-3}$

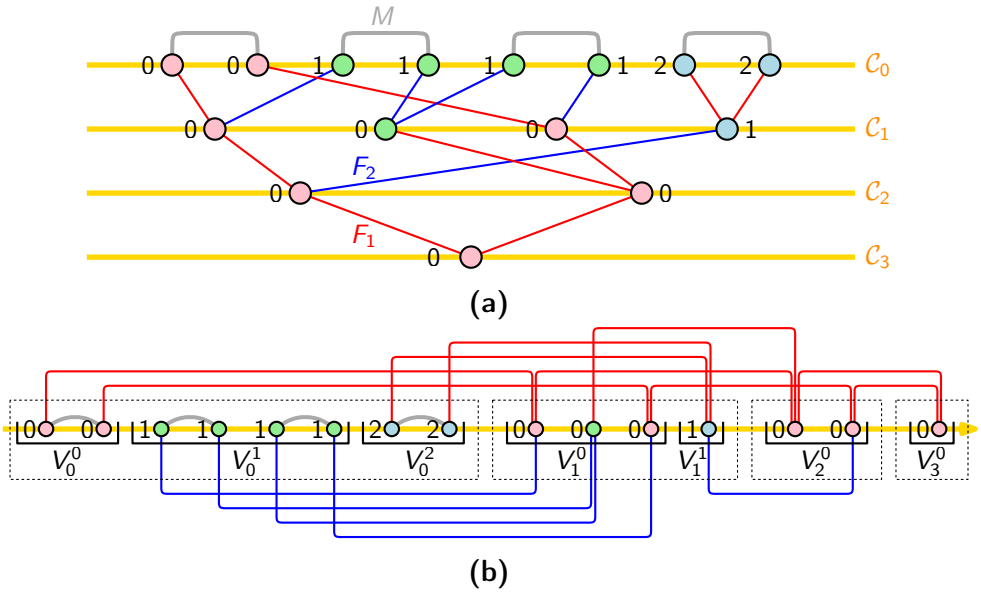


Figure 5.5: (a) The 3-matched graph G from Figure 5.4 where the vertices on each layer have been reordered according to $\prec$ as computed by our algorithm. Matching edges (gray) form a necklace while the edges of T can be separated into two forest F_1 (red) and F_2 (blue) which are intersection-free because $\prec$ is a good tree ordering for both forests. Vertices are labeled with their layer group and colored in correspondence with Figure 5.4. (b) Resulting 3-page queue layout.

like this; for an illustration refer to Fig. 5.5a. We now show that $\prec$ is a good tree ordering for each of forests $F_1, \ldots, F_{2\Delta-3}$ (which in Fig. 5.5a is reflected by the fact that all forests are drawn planar). Lemma 5.1 then implies that there exists a $(2\Delta-2)$-page queue layout where each of the forests $F_1, \ldots, F_{2\Delta-3}$ is located on a unique page while M is located on a separate page $\mathcal{P}_M$.

Consider one of the forests F_i. We first observe that $\prec$ fulfils Property T.1 since the layering of G is according to the graph-theoretic distance to the root r of T. Hence, we only have to assert Property T.2.

For this purpose, consider two edges (u, v) and (u', v') assigned to forest F_i such that $\ell(u) = \ell(u') = \ell(v) + 1 = \ell(v') + 1$ and $u \neq u'$. We have two subcases: If w.l.o.g. $g(u) < g(u')$, it holds by Condition C.2 that $u \prec u'$. Since u and u' belong to the same forest F_i, it holds that $g(v) = g(u)(\Delta-1) + i < g(u')(\Delta-1) + i = g(v')$ and by Condition C.2, it follows that $v \prec v'$. Hence, in this case, Property T.2 holds.

Otherwise $g(u) = g(u')$. Since u and u' belong to the same forest F_i, it holds that $g(v) = g(u)(\Delta-1) + i = g(u')(\Delta-1) + i = g(v')$. Assume w.l.o.g. that $u \prec_{\ell(u)} u'$. By Property P.3, it follows that $v \prec_{\ell(v)} v'$. Then by Condition C.3 also $u \prec u'$ and $v \prec v'$ which fulfils Property T.2. We conclude that $\prec$ is indeed a good tree ordering for forest F_i and the proof follows. □

5.3 General Planar Graphs of Bounded Degree

In this section, we use the result from the previous section to show that the queue number of the class of bounded degree planar graphs is bounded. Before we go into detail, we sketch our approach to compute a queue layout of a planar graph G with maximum degree Δ:

1. As a first step, we subdivide some edges of G at most three times to obtain a graph G_1 which consists of a BFS-tree T rooted at a vertex r and some level edges regarding T.

2. In a second step, we replace the remaining level edges and their incident vertices with appropriate Δ-matched subgraphs. The resulting graph G_2 then is Δ-matched.

3. In the third step, we first apply Lemma 5.4 to obtain a $(2\Delta - 2)$-page queue layout of G_2. We then observe that the level edges that were replaced when building G_2 based on G_1 can be realized as a necklace which yields a $(2\Delta - 2)$-page queue layout of G_1. Finally, we use Lemma 5.2 in order to obtain Theorem 5.1.

In the following, we assume w.l.o.g. that $\Delta \geq 3$ as all graphs with maximum degree at most two have queue number one [108] which settles Theorem 5.1 in the case where $\Delta \leq 2$. In addition assume w.l.o.g. that G has no vertex v with $deg(v) = 1$. Otherwise, we can insert vertices v_1 and v_2 and edges (v, v_1), (v, v_2) and (v_1, v_2) for each vertex v with $deg(v) = 1$ to obtain a graph G' without vertices of degree one. Since the queue number of G is at most the queue number of G', this assumption is indeed not a loss of generality.

Step 1: Construction of G_1. We begin by selecting a vertex r of G with $deg(r) = 2$. If no such vertex exists, we instead subdivide an arbitrary edge once obtaining a degree two vertex r. We then compute an ordered concentric representation R of G centered at r. Let $\mathcal{C}_0, \ldots, \mathcal{C}_h$ denote the concentric circles of R ordered decreasing in their radius and let T be the BFS-tree rooted at R that we used for computing R according to Section 5.1; see also Fig. 5.6a. Recall that R contains two types of edges: *binding edges* (i.e., edges that are incident to vertices at consecutive circles), and *level edges* (i.e., edges between vertices on the same circle). Here, we further distinguish two types of binding edges, namely, *tree edges* (i.e., edges that belong to T) and *interlevel edges* (i.e., edges that do not belong to T). Note that if r is a subdivision vertex created by subdividing edge (u, v) the two resulting edges (u, r) and (r, v) are tree edges. We will not subdivide tree edges again.

We obtain G_1 in two steps. First, we subdivide each interlevel edge (u, v) with $\ell(v) > \ell(u)$ once with subdivision vertex v'. In the resulting graph, edge

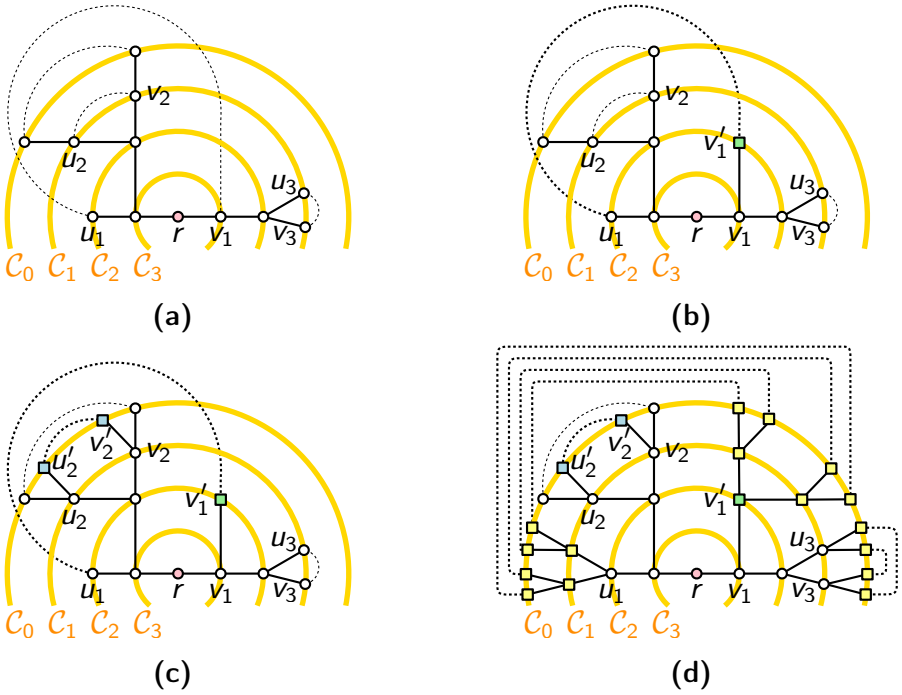


Figure 5.6: An example of the computations of auxiliary graphs G_1 and G_2. Tree edges are drawn solid in all subfigures. (a) Ordered concentric representation of G with root r. (b) Interlevel edge (u_1, v_1) where $\ell(v_1) > \ell(u_1)$ is replaced by path (u_1, v_1', v_1) such that $\ell(v_1') = \ell(u_1)$. The resulting path consists of a tree edge and a level edge. (c) Level edge (u_2, v_2) is replaced by path (u_2, u_2', v_2', v_2) which consists of two tree edges and a level edge incident to the two degree two leaves u_2' and v_2'. The resulting graph is G_1. (d) Level edges (u_3, v_3) and (u_1, v_1') are replaced by trees rooted at their endpoints and matching edges obtaining graph G_2.

(v, v') is a tree edge while edge (u, v') is a level edge. For instance interlevel edge (u_1, v_1) in Fig. 5.6a is being subdivided obtaining the graph shown in Fig. 5.6b. Second, we subdivide each level edge (u, v) twice with subdivision vertices u' and v' if $deg(u) > 2$ or $deg(v) > 2$. The resulting edges (u, u') and (v, v') are tree edges while (u', v') is a level edge. For instance, edge (u_2, v_2) in Fig. 5.6b becomes subdivided twice yielding the graph shown in Fig. 5.6c. The resulting 3-subdivision of G is the auxiliary graph G_1. Note that all level edges in G_1 are incident to two leaves of the BFS-tree T_1 of G_1.

Step 2: Construction of G_2. Graph G_1 is already almost a Δ-matched graph, in fact, only Property P.1 is violated since some of the leaves of the BFS-tree T_1 of G_1 are not on $\mathcal{C}_0$. We apply the following procedure to each level edge (u, v) on $\mathcal{C}_\ell$ with $\ell > 0$. We remove (u, v) and insert two complete $(\Delta - 1)$-ary trees T_u and T_v of height ℓ rooted at vertices u and v, respectively. Then, we insert a perfect matching $M_{u,v}$ of the leaves on T_u and T_v such that the i-th leaf of T_u is

matched with the $((\Delta-1)^\ell - i)$-th leaf of T_v according to the ordering $\prec_0$. As a result, $M_{u,v}$ is forming a $(\Delta-1)^\ell$-rainbow in $\prec_0$. For an illustration refer to Fig. 5.6d which is obtained from the drawing in Fig. 5.6c by replacing level edges (u_3, v_3) and (u_1, v_1'). Graph G_2 is indeed planar as we can insert trees T_u and T_v and matching edges $M_{u,v}$ following the representation of (u, v) in G_1. Trees T_u and T_v become part of the BFS-tree T_2 of G_2. Since after this operation all leaves of T_2 are located on $\mathcal{C}_0$ while all remaining edges are part of a planar matching on $\mathcal{C}_0$, it follows that G_2 indeed is a Δ-matched graph. Moreover, T_2 defines an ordered concentric representation with the properties described in Property P.2.

Step 3: Construction of a queue layout of G. We begin by computing a $(2\Delta-2)$-page queue layout Γ_2 of G_2 using Lemma 5.4.

Next, we create a queue layout Γ_1 of G_1 based on Γ_2. All edges of G_1 that also occur in G_2 will be drawn the same way, hence, it remains to consider level edges (u, v) on layer $\ell > 0$ that were replaced by T_u, T_v and $M_{u,v}$ in G_2. We claim the following:

Lemma 5.5. *Let L be the set of level edges (u, v) on layer $\ell > 0$ that were replaced by T_u, T_v and $M_{u,v}$ in G_2. We can draw all edges in L on page $\mathcal{P}_M$ using the vertex ordering $\prec$ of Γ_2.*

Proof. Assume for a contradiction that there are two such edges (u, v) and (u', v') such that (u, v) nests (u', v'), i.e., $u \prec u' \prec v' \prec v$. By Condition C.1 of $\prec$, it follows that $\ell(u) = \ell(v) = \ell(u') = \ell(v')$, i.e., both edges occur on the same layer ℓ. Next consider the matching values $mv(u)$, $mv(v)$, $mv(u')$ and $mv(v')$. Since the matching edges incident to subtrees T_u and T_v in G_2 are identical, it holds that $mv(u) = mv(v)$ and therefore also $g(u) = g(v)$. Similarly, $mv(u') = mv(v')$ and $g(u') = g(v')$. If $g(u) = g(v) < g(u') = g(v')$, by Condition C.2, it holds that $v' \not\prec v$ which contradicts our assumption. Hence, $g(u) = g(v) = g(u') = g(v')$. Since the ordered concentric representation of G_1 is planar, there are two possible cases. Either edges (u, v) and (u', v') are independent with respect to $\prec_\ell$ or (u, v) and (u', v') nest with respect to $\prec_\ell$. By Condition C.3, it follows that edges (u, v) and (u', v') show the same behavior in $\prec$ as they show in $\prec_\ell$. Therefore, (u, v) must nest (u', v') in $\prec_\ell$. But then, all of the $(\Delta-1)^\ell$ edges of $M_{u,v}$ nest all edges of $M_{u',v'}$ in $\prec_0$ of the ordered concentric representation of G_2. Since vertices u and u' belong to layer ℓ, it follows that $mv(u') \geq mv(u) + (\Delta-1)^\ell$ and thus by Eq. (5.1) also $g(u') > g(u)$; a contradiction. □

We conclude that indeed the edges assigned to $\mathcal{P}_M$ form a necklace in $\prec$. Thus, we have a $(2\Delta-2)$-page queue layout Γ_1 of G_1. Since we obtained G_1 from G by subdividing each edge at most three times, we can apply Lemma 5.2 with $k = 3$ and $q = 2\Delta-2$ and obtain:

Theorem 5.1. *Let G be a planar graph of maximum degree Δ. Then, it holds that $qn(G) \leq 32(2\Delta-1)^6 - 1$.*

5.4 Time Complexity

In this section, we analyze the runtime of our algorithm that constructs a $(32(2\Delta-1)^6-1)$-page queue layout for planar graphs of maximum degree Δ. We first reevaluate our algorithm for Δ-matched graphs:

Lemma 5.6. *Let G be a Δ-matched graph with an ordered concentric representation R with the properties described in Property P.2. Then a k-page queue layout of G with $k \leq 2\Delta-2$ can be computed with $\mathcal{O}(n)$ basic computations (additions, comparisons) that involve a constant number of values each.*

Proof. We consider the runtime of each of the steps performed by the algorithm in the proof of Lemma 5.4:

1. The nesting value for each edge in M is computed. Given the ordered concentric representation of G, this can be performed with $\mathcal{O}(|M|) = \mathcal{O}(n)$ basic computations.

2. The matching value for each vertex in G is computed. Based on the nesting value of each edge in M, the matching values of leaves can be computed with $\mathcal{O}(|M|)$ basic computations. For the remaining vertices, a bottom-up traversal of T yields the corresponding matching values. Hence, we compute all matching values with $\mathcal{O}(n)$ basic computations.

3. The layer group of each vertex v can be easily deduced from the matching value $mv(v)$ which was computed before and the layer $\ell(v)$ that is defined by R. Hence, all layer groups can be computed with $\mathcal{O}(n)$ basic computations.

4. The linear order of vertices can be computed based on a single traversal through each of the linear orders $\prec_\ell$ for $0 \leq \ell \leq h$ which are defined by R; see Conditions C.1–C.3. Hence, this step can be done with $\mathcal{O}(n)$ basic computations.

5. The assignment of edges to pages of the queue-layout are determined on the layer groups of their endpoints which were already computed. Hence, for each edge, we only have to look up two values. In total, we need $\mathcal{O}(m) = \mathcal{O}(n)$ basic computations.

We conclude that the algorithm in the proof of Lemma 5.4 needs $\mathcal{O}(n)$ basic computations. □

The time critical step in the computation of a queue layout of a general planar graph G of maximum degree Δ is Step 2, that is, the construction of G_2. Namely, for a level edge (u, v) on layer ℓ, we introduce new trees T_u and T_v on $\Delta^{\Theta(\ell)}$ vertices. Note that ℓ is bounded by the height h of the BFS-tree T_1 of G_1.

However, a BFS-tree of a planar graph may have height $\Theta(n)$. So in order to compute a queue layout of G efficiently, we cannot explicitly introduce such trees. Instead, we assign weights to level edges of G_1 and compute a $(2\Delta - 2)$-page queue layout of G_1 directly.

In fact, the purpose of such trees is the following: Consider two level edges (u, v) and (u', v') on layer ℓ. If (u, v) nests (u', v') in $\prec_\ell$, replacing (u, v) with T_u, T_v and $M_{u,v}$ ensures that $g(u) = g(v) > g(u') = g(v')$. This is because each of the edges in $M_{u,v}$ nests each edge of $M_{u',v'}$ in $\prec_0$ while $|M_{u,v}| = (\Delta - 1)^\ell$. We can achieve the same effect as follows: Instead of replacing each level edge (u, v) with T_u, T_v and $M_{u,v}$ we route it outside of layer 0. If we perform this operation first on edges on layers with lower indices, this can be done in a single pass of the level edges on each of the layers. Then, we introduce a *weight* $w(u, v)$ equal to $|M_{u,v}|$. Since the new routing of edge (u, v) corresponds to the routing of T_u, T_v and $M_{u,v}$, the piece of (u, v) outside of $\mathcal{C}_0$ nests everything that $M_{u,v}$ would have nested before. Moreover, if we perform this operation on all level edges, each level edge has a piece outside of $\mathcal{C}_0$.

We then update the computation of nesting values and matching values in Step 3 as follows: Edge (u, v) has nesting value 0 if and only if its piece outside of $\mathcal{C}_0$ is not nested by any piece of another edge (u', v') outside of $\mathcal{C}_0$. Otherwise, let (u', v') be the edge of maximum nesting value which has a piece outside of $\mathcal{C}_0$ that nests the piece of (u, v). Then, we set $n((u, v)) = n((u', v')) + w(u', v')$. As a result, each vertex of G_1 is assigned the same matching value as it would have been assigned if we replaced level edges as discussed before. We point out that the largest nesting value is in $\mathcal{O}(\Delta^n)$ since the height of the BFS-tree is bounded by the number of vertices. Hence, we can represent all nesting and matching values with a $\mathcal{O}(n \log \Delta)$ length binary string. Since we only use standard operations on such values (namely, addition and comparison), each such operation can be performed in time $\mathcal{O}(n \log \Delta)$.

Based on the matching and nesting values we can then compute a $(2\Delta - 2)$-page queue layout using the linear order defined by Conditions C.1–C.3. As a result, we obtain the following:

Theorem 5.2. *Let G be a planar graph of maximum degree Δ. Then, a k-page queue layout of G with $k \leq 32(2\Delta-1)^6-1$ can be computed in time $\mathcal{O}(n^2 \log \Delta)$.*

Proof. We reconsider the steps of the algorithm supporting Theorem 5.1:

1. In order to construct G_1 in Step 1, we compute an ordered concentric representation R which can be done in time $\mathcal{O}(n)$ by Lemma 5.3. Then, we subdivide all interlevel edges in time $\mathcal{O}(n)$ since they are uniquely defined by R. Finally, we obtain G_1 by splitting each of the level edges at most twice. Those edges are again well-defined by R and can be found in time $\mathcal{O}(n)$.

2. We use the new version of Step 2 as described above. Namely, we reroute each of the level edges of G_1 which can be done by first rerouting edges on

layers with lower indices. Hence, we traverse each layer once achieving time $\mathcal{O}(n)$ in total. Then, we assign weights to each level edge. As discussed above, the weights of level edges can be represented with $\mathcal{O}(n \log \Delta)$ bits each, hence, this takes time $\mathcal{O}(n^2 \log \Delta)$.

3. In Step 3, we first compute the nesting and matching values as described above. For nesting values, we have to make a pass over all level edges and in each step deal with a constant number of values representable by $\mathcal{O}(n \log \Delta)$ bits each. Hence, this takes time $\mathcal{O}(n^2 \log \Delta)$ time. Then, we calculate the matching values in a bottom-up traversal during which $\mathcal{O}(n)$ calculations are performed. As a result, all matching values can be computed in time $\mathcal{O}(n^2 \log \Delta)$. We then use the remaining steps from Lemma 5.4 to compute a $(2\Delta - 2)$-page queue layout of G_1. Since each basic computation may involve numbers of size $\mathcal{O}(n \log \Delta)$ in bit representation, this may take time $\mathcal{O}(n^2 \log \Delta)$. Finally, we use Lemma 5.2 to compute the queue layout of G. While the time complexity of Lemma 5.2 is not explicitly stated in [81], it is easy to see that it can be performed in time $\mathcal{O}(n)$ when there are only a constant number of subdivision vertices per edge. In particular, the main operation performed in the lemma is the computation of a track layout of a subgraph of G [79] which is repeated a number of times which is logarithmic in the constant number of subdivision vertices.

Since we proved that each of the three steps can be performed in total time $\mathcal{O}(n^2 \log \Delta)$, the theorem follows. □

Chapter 6

Monotone Arc Diagrams with few Biarcs

Only subhamiltonian planar graphs, that is, subgraphs of planar graphs with Hamiltonian cycles, admit a 2-page stack layout [40]. In general, up to four pages may be required for visualizing planar graphs in a stack layout [30, 161]. However, even 3-page stack layouts admit no nice embedding in the plane, since two pages have to use the same half-plane delimited by the spine which can introduce many intersections.

A different approach to visualize graphs with vertices restricted to a line are the so-called arc diagrams in which edges are realized on two pages, i.e., a half-plane below and a half-plane above the spine. In order to realize all planar graphs, some edges have to be drawn as biarcs, that is, a sequence of a segment above and a segment below the spine. In fact, all biarcs can even be down-up monotone, i.e., monotone with respect to the spine such that the left segment is below the spine.

Arc diagrams find applications in circular layouts [65] and point set embeddability problems, in particular, for the cases where edges are drawn as circular arcs [15] as well as 1-bend polylines [89, 130]. In the latter application, specifically down-up monotone arc diagrams are required. In either scenario, the number of biarcs in the arc diagram affects the quality of the drawing: In circular layouts, biarcs become curves that cross the circle to which vertices are restricted, while the number of points in universal point sets is tied to the number of biarcs that may be required in an arc diagram. As a result, the number of required biarcs has been investigated in the literature [51, 67, 120]. The best known upper bounds are $\lfloor (n-3)/2 \rfloor$ for general plane arc diagrams and $n-4$ for plane down-up monotone arc diagrams [51]. The lower bound for the required number of biarcs on the other hand is $\lfloor (n-8)/3 \rfloor$ in both the down-up monotone and general case.

It is noteworthy, that monotone arc diagrams are not as well understood as general arc diagrams while they are specifically required in some applications. In

this chapter[1], we provide more insight in this regard by showing the following:

Theorem 6.1. *Every planar graph admits a plane down-up monotone arc diagram with at most* $\lfloor 15/16n - 5/2 \rfloor$ *biarcs that can be computed in linear time.*

As a side result we describe a SAT formulation, that is based on [39], with which we verified the following:

Observation 6.1. *No Kleetope based on a triangulation with up to* 14 *vertices requires more biarcs in its down-up monotone arc diagrams compared to its general arc diagrams even if the outer face is arbitrarily prescribed.*

In the remainder of the chapter, we first describe an algorithm that asserts Theorem 6.1 in Sections 6.1 to 6.4. More precisely, in Section 6.1, we provide an overview of our algorithm that inserts vertices one at a time. Here, we also introduce the distinction between *default* steps of the algorithm and more complex steps in which vertices are inserted above so-called *open configurations*. Then, we discuss the default steps in Section 6.2 and the steps involving open configurations in Section 6.3. We summarize the proof of Theorem 6.1 in Section 6.4. Afterwards, we shift our attention to the SAT formulation in Section 6.5.

6.1 Overview of the Algorithm

Our algorithm is an elaborate improvement of the algorithm by Cardinal et al. [51]. We assume w.l.o.g. that the input graph of our algorithm is a triangulation; otherwise we can triangulate it. We insert the n vertices $v_1, \ldots, v_n$ according to a canonical ordering. Recall that G_i denotes the graph consisting of vertices $v_1, \ldots, v_i$ while C_i is the outer face of the drawing of G_i. In addition, $deg_{G_i}(v)$ denotes the degree of vertex v in G_i.

We will pay close attention to the part of the outer face C_{i-1} of the drawing of graph G_{i-1} that is *covered* by vertex v_i. More formally, we say that v_i covers an edge e (or vertex v, respectively) if and only if e (v, respectively) is an edge (vertex, respectively) on C_{i-1} but not on C_i. In addition, we distinguish two types of proper arcs, namely *mountains* (above the spine) and *pockets* (below the spine).

Every edge that is not a proper arc, will be realized as a down-up biarc. In order to bound the number of biarcs created by our algorithm, we require that one *credit* is allocated to each biarc. When inserting vertex v_i, we also introduce α credits to the drawing that can either be allocated to biarcs or stored on edges whose endpoints are on the outer face C_i so that they can be used later.

The following invariants hold after inserting vertex v_i for the drawing of G_i:

I.1 Every edge is either drawn as a proper arc or as a down-up biarc.

[1]The results of this chapter also appeared in [52].

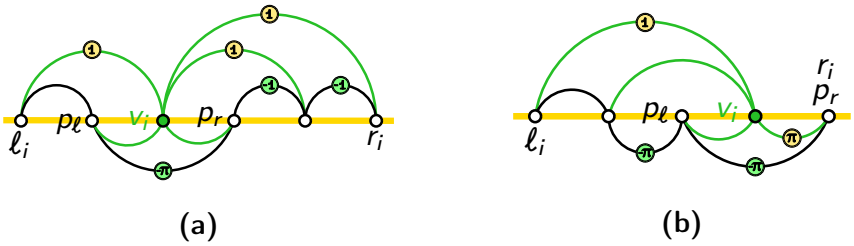


Figure 6.1: Inserting a vertex v_i onto a pocket costs at most $1 - \pi$ credits if $deg_{G_i}(v_i) \geq 4$: (a) If $r_i \neq p_r$, the credit from pocket (p_ℓ, p_r) reduces the cost to at most $1 - \pi$. (b) If $r_i = p_r$, at least one pocket (or mountain) left of p_ℓ is covered.

I.2 Let $w_1 = v_1, w_2, \ldots, w_{p-1}, w_p = v_2$ denote the vertices of C_i ordered from left to right along the spine. Then, edge (v_1, v_2) forms the lower envelope of the drawing of G_i while the path $(w_1, w_2, \ldots, w_{p-1}, w_p)$ forms the upper envelope. All edges of C_i are drawn as proper arcs.

I.3 Every mountain whose left endpoint is part of C_i is allocated 1 credit.

I.4 Every pocket on the outer face C_i is allocated π credits, for some constant $\pi \in [0, 1)$.

I.5 Every biarc in G_i is allocated 1 credit.

In most cases, we do not discuss explicitly that we maintain the invariants. We visualize the assignment of credits to edges in the figures with coin symbols π and 1 for π and a full credit, respectively. In addition, if we cover an edge that was assigned credits in G_{i-1} but not in G_i, we visualize that some unused credit can be reallocated with green coin symbols -π and -1 for π and a full credit, respectively.

In most cases, we will insert vertex v_i above one of the edges it covers, i.e., between its *leftmost neighbor* ℓ_i and its *rightmost neighbor* r_i on C_{i-1}. The following two lemmas are directly derived from the two different cases (some covered edge is a pocket or no covered edge is a pocket) that can arise in the algorithm by Cardinal et al. [51] and give a first bound for α.

Lemma 6.1. *If at least one pocket is covered by v_i, vertex v_i can be inserted for a cost $c \leq 1$ such that Invariants I.1 to I.5 hold. If $deg_{G_i}(v_i) \geq 4$, then $c \leq 1 - \pi$.*

Proof. Vertex v_i is always placed onto the rightmost covered pocket (p_ℓ, p_r) while all incident edges are drawn as proper arcs; see Fig. 6.1. In particular, edges (p_ℓ, v_i) and (v_i, p_r) are drawn as pockets while all remaining new edges are drawn as mountains which satisfies Invariants I.1 and I.2.

If (v_i, p_r) is not covered, we reallocate the π credits from (p_ℓ, p_r) to it to maintain Invariant I.4. Otherwise, consider a vertex r to the right of p_r incident to v_i. Edge (v_i, r) is a mountain and requires one allocated credit by Invariant I.3.

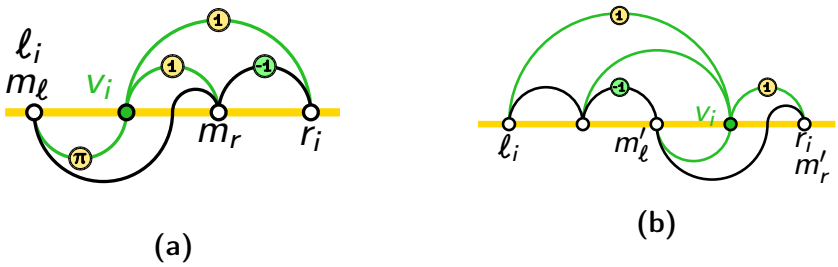


Figure 6.2: (a) Inserting a vertex v_i on top of mountains only costs at most $1+\pi$ credits. (b) If $deg_{G_i}(v_i) \geq 4$, the cost can be reduced to $5 - deg_{G_i}(v_i)$.

On the other hand, since (p_ℓ, p_r) is the rightmost pocket, the left neighbor of r on C_{i-1} is connected to r with a mountain m_r. By Invariant I.3, m_r has a credit which we can reallocate to (v_i, r).

In addition edge (ℓ_i, v_i) may be either realized as a mountain or as a pocket. If it is realized as a mountain, we have to allocate one credit on it by Invariant I.3, while if it is realized as a pocket, we have to allocate π credits on it by Invariant I.4. It might not be possible to redistribute these credits from covered edges, i.e., $c \leq 1$. This suffices to prove the part of the lemma where $deg_{G_i}(v_i) < 4$.

If $r_i \neq p_r$, pocket (v_i, p_r) is not on the outer face and hence does not need to carry π credits. Then, the π credits from edge (p_ℓ, p_r) can be reallocated and $c \leq 1 - \pi$; see Fig. 6.1a. If $r_i = p_r$ and $deg_{G_i}(v_i) \geq 4$, there exists at least one covered edge e on the outer face whose right endpoint is p_ℓ that we did not consider so far. Since the left endpoint of e is covered by v_i, we can reallocate its credits; see Fig. 6.1b. By Invariants I.3 and I.4, e was allocated at least π credits in C_{i-1} and we can insert v_i for $c \leq 1 - \pi$. □

So far, we discussed the case where v_i covers a pocket. If this is not the case, vertex v_i has no access to the spine between its leftmost neighbor ℓ_i and its rightmost neighbor r_i and biarcs have to be created. In such a case, one mountain m of the outer face will be *pushed down*, that is, m and all mountains that have the same left endpoint as m are transformed into down-up biarcs. By Invariant I.3 each of those mountains carries a credit so pushing down m does not violate Invariant I.5. The following lemma estimates the cost for inserting v_i when covering only mountains:

Lemma 6.2. *If only mountains are covered by v_i, vertex v_i can be inserted for a cost $c \leq 1 + \pi$ such that Invariants I.1 to I.5 hold. If $deg_{G_i}(v_i) \geq 4$, the cost is $c \leq 5 - deg_{G_i}(v_i)$.*

Proof. First, consider the case where $deg_{G_i}(v_i) \leq 3$. Here, we push down the leftmost mountain (m_ℓ, m_r) and place v_i between m_ℓ and m_r and realize edge (m_ℓ, v_i) as a pocket and all other edges incident to v_i as a mountain; see Fig. 6.2a.

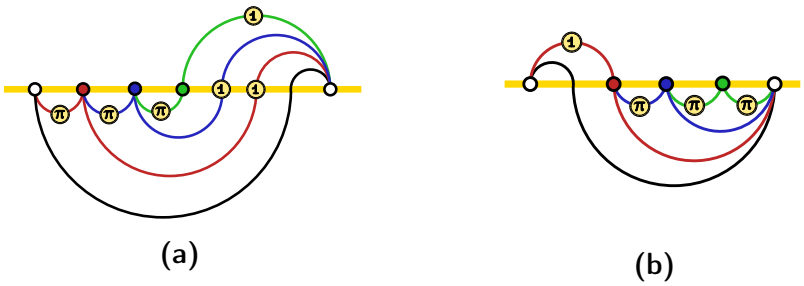


Figure 6.3: A repeatable configuration that (a) creates many biarcs when using only down-up biarcs and (b) creates only one biarc when using only up-down biarcs.

The credits for newly created biarcs are already allocated to the corresponding edges by Invariant I.3. In order to satisfy Invariant I.4, we have to allocate π credits to (m_ℓ, v_i). In addition, we have to allocate 1 credit to each mountain incident to v_i. We have to pay the credit for mountain (v_i, m_r). If $r_i \neq m_r$, we have $deg_{G_i}(v_i) = 3$ and we also cover mountain (m_r, r_i) and can reallocate the credit to the new mountain (v_i, r_i); see Fig. 6.2a. Therefore, $c \leq 1 + \pi$.

Second, consider the case where $deg_{G_i}(v_i) \geq 4$. Here, we instead push down the rightmost mountain (m'_ℓ, m'_r) and place v_i between m'_ℓ and m'_r and realize edge (m'_ℓ, v_i) as a pocket and all other edges incident to v_i as a mountain; see Fig. 6.2b. Here, we have to allocate a credit to new mountains (ℓ_i, v_i) and (v_i, r_i). However, there are at least $deg_{G_i}(v_i) - 3$ more covered mountains whose left endpoints got covered as well. When redistributing the credits of these mountains, we achieve a cost of $2 - (deg_{G_i}(v_i) - 3) = 5 - deg_{G_i}(v_i)$. □

It is noteworthy that in the proofs of Lemmas 6.1 and 6.2, we only reallocate credits from arcs on C_{i-1}. However, if we cover the left endpoint of a mountain not on C_{i-1} there may be some slack. Also observe that so far we can only say that $\alpha \leq 1 + \pi$. By setting $\pi = 0$, we obtain the result from Cardinal et al. [51]. In order to reduce the number of biarcs, we want to choose $\pi > 0$ to improve the result from Lemma 6.1. This will prove useful as in the critical case of Lemma 6.2, we create new pockets whose π credits may be reallocated in a future step.

We also point out that it is impossible that the case $deg_{G_i}(v_i) < 4$ continually occurs due to the density of maximal planar graphs. However, it is possible that a sequence of degree two vertices that becomes stacked on top of mountains can occur in the canonical ordering; see Fig. 6.3a. In such a case, for every inserted vertex a biarc is created and the worst-case cost of $1 + \pi$ is achieved. However, if we would adopt a symmetric scheme with up-down biarcs, we would only end up with one created biarc for the first inserted vertex; see Fig. 6.3b.

In order to use this behavior, we actually create two drawings, called *forward* (where we only use down-up biarcs) and *reverse* (where we only use the symmetric up-down biarcs). In the forward drawing $\overrightarrow{\Gamma}_i$ of G_i, we use the version of Invari-

ants I.1 and I.3 as stated before while in the reverse drawing $\overleftarrow{\Gamma}_i$ of G_i we use symmetric formulations. More precisely, in $\overleftarrow{\Gamma}_i$, we use up-down biarcs instead of down-up ones while mountains whose right endpoints (in contrast to the left ones) are part of C_i are allocated 1 credit. From the two resulting drawings, we choose the one with fewer biarcs after inserting all vertices. Note that C_i can be drawn differently in $\overrightarrow{\Gamma}_i$ and $\overleftarrow{\Gamma}_i$ which will give rise to many different cases later.

As mentioned before, we want to claim later that on average we can save the credits allocated to one pocket of the outer face. Hence, we will show in the remainder of the description of the algorithm that we can insert vertices at an average cost $\alpha = 2 - \pi$. Unfortunately, this may not be the case for a single vertex that is considered in isolation: Consider a vertex v_i of degree three that is inserted above two mountains in both $\overrightarrow{\Gamma}_{i-1}$ and $\overleftarrow{\Gamma}_{i-1}$; see Fig. 6.4b. In both drawings, we tightly achieve the cost of $1 + \pi$ credits as stated in Lemma 6.2. Hence, we pay $2 + 2\pi$ for both drawings, which is 3π credits more than α. In fact, it is not possible to improve this result here when placing v_i between ℓ_i and r_i which however is necessary to maintain Invariant 2.

We solve this problem by introducing *open configurations*. An open configuration $\mathcal{C}$ consists of up to two adjacent vertices c_1 and possibly c_2 on the outer face and their incident edges where the insertion of c_1 and possibly c_2 could not be covered by the α or possibly 2α credits introduced when inserting c_1 and possibly c_2. Each open configuration $\mathcal{C}$ is associated with a debt $d(\mathcal{C})$ which is the amount of additional credits that were required to maintain Invariants 1 to 5 when inserting c_1 and possibly c_2. Once any arc of $\mathcal{C}$ becomes covered by a vertex v_i, we either have to cover the debt of $\mathcal{C}$ or create a new open configuration $\mathcal{C}'$ that consists of the vertex of $\mathcal{C}$ and v_i. Note that after inserting v_n, we have at most one open configuration left. We formulate this new property as another invariant:

I.6 The outer face C_i may contain pairwise disjoint open configurations consisting of up to two adjacent vertices and their incident edges. An open configuration $\mathcal{C}$ is associated with a debt $d(\mathcal{C}) \leq 5\pi$.

We list all possible types of open configurations labeled $\mathcal{C}_a, \ldots, \mathcal{C}_k$ in Fig. 6.4.

We point out that Lemmas 6.1 and 6.2 do only account for the costs of paying the edges incident to v_i while ignoring the debts of possibly existing open configurations. In the following section, we discuss how we process vertices that do not cover arcs belonging to open configurations; here we will introduce most of the open configurations listed in Fig. 6.4. Then, in Section 6.3, we consider the case where vertices are inserted that cover at least one arc of an open configuration. In either occuring case, we will show that α credits are sufficient if $\pi \leq 1/8$.

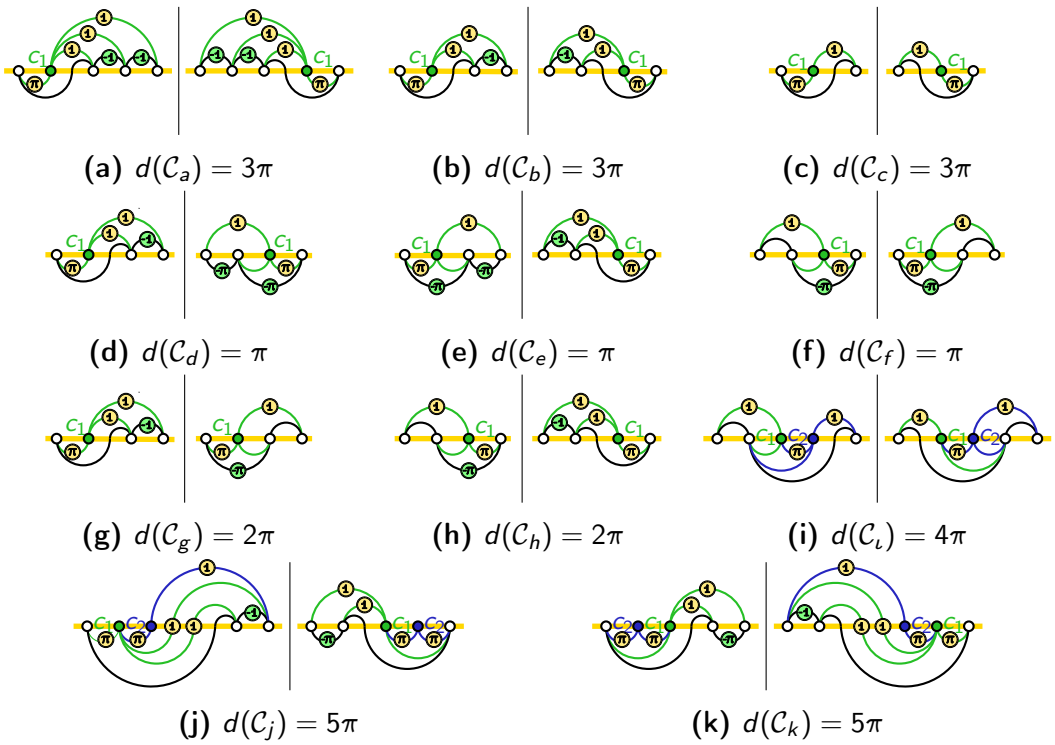


Figure 6.4: All eleven types of open configurations $\mathcal{C}_a, \ldots, \mathcal{C}_k$ that can be created. In each subfigure, we show the forward (left) and the reverse drawing (right) while the caption shows the corresponding debt. Note that pairs $(\mathcal{C}_d, \mathcal{C}_e)$, $(\mathcal{C}_g, \mathcal{C}_h)$ and $(\mathcal{C}_j, \mathcal{C}_k)$ are symmetric while the remaining open configurations are symmetric to themselves.

6.2 Default Vertex Insertion

In this section, we discuss how we insert a vertex v_i if it does not cover any arc that is part of an open configuration. In principal, we apply the procedures discussed in Lemmas 6.1 and 6.2. If the cost exceeds the α credits introduced with v_i, we introduce an open configuration.

If $deg_{G_i}(v_i) \geq 4$ and at least one pocket is covered in either of the two drawings, by Lemmas 6.1 and 6.2 the sum of both insertion costs is at most $2 - \pi = \alpha$. If we however only cover mountains and $deg_{G_i}(v_i) = 4$, we end up with the open configuration $\mathcal{C}_a$ shown in Fig. 6.4a that has a cost of $2 + 2\pi$ and hence a debt of 3π. Note that by Lemma 6.2, we can achieve a cost of 2 for $\mathcal{C}_a$, however we do not use the cheapest drawing to obtain a pocket on the outer face which becomes easier to handle later. On the other hand, if $deg_{G_i}(v_i) \geq 5$, by Lemma 6.2 we achieve a total cost of 0 for both drawings if we only cover mountains.

Next, consider the case where $deg_{G_i}(v_i) = 2$. If v_i covers a pocket in either of the two drawings, the insertion in this drawing costs only π. Therefore, we achieve a total cost of at most $1 + 2\pi$ which is at most α if $\pi \leq 1/3$. If we cover a mountain in both drawings however, we obtain the open configuration $\mathcal{C}_c$ shown in Fig. 6.4c that has a cost of $2 + 2\pi$ and hence a debt of 3π.

Finally, consider the case where $deg_{G_i}(v_i) = 3$. Here, there are four possible configurations for the two arcs that are covered by v_i, namely from left to right: mountain-mountain (or MM), mountain-pocket (or MP), pocket-mountain (or PM) and pocket-pocket (or PP). In either drawing, we can insert v_i on the configuration PP for a cost of $1 - \pi$ while we can insert v_i on the configuration MM for $1 + \pi$. The insertion into PM costs 0 in the forward and 1 in the reverse drawing, while the insertion into MP costs 1 in the forward and 0 in the reverse drawing. We consider all possible *forward*|*reverse* configuration combinations:

- Insertion into $MM|MM$ costs $2 + 2\pi$ and we obtain the open configuration $\mathcal{C}_b$ shown in Fig. 6.4b with debt of 3π.
- Insertion into $MM|PM$ costs $2 + \pi$ and we obtain the open configuration $\mathcal{C}_g$ shown in Fig. 6.4g with debt of 2π. Symmetrically, if we insert into pattern $MP|MM$, we obtain the open configuration $\mathcal{C}_h$ shown in Fig. 6.4h.
- Insertion into $MM|PP$ costs 2 and we obtain the open configuration $\mathcal{C}_d$ shown in Fig. 6.4d with debt of π. Symmetrically, if we insert into pattern $PP|MM$, we obtain the open configuration $\mathcal{C}_e$ shown in Fig. 6.4e.
- Insertion into $MP|PM$ costs 2 and we obtain the open configuration $\mathcal{C}_f$ shown in Fig. 6.4f with debt of π.
- Insertion into all other combinations costs at most α for any π.

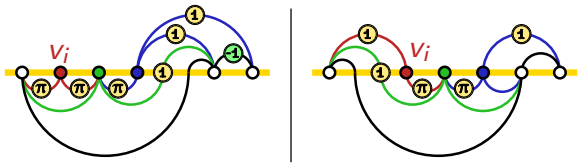


Figure 6.5: Drawing for the insertion of a degree two vertex v_i that covers a mountain belonging to an open configuration of type $\mathcal{C}_\iota$.

Hence, if v_i covers no arc of an open configurations, we can always insert it such that the α new credits are sufficient for reallocating credits in both drawings or we end up with one of the open configurations listed in Fig. 6.4.

6.3 Vertex Insertion involving Open Configurations

In this section, we discuss how we insert a vertex v_i if it covers at least one arc that is part of an open configuration. Before we begin, we point out two properties of open configurations that will become useful in the discussion; see also Fig. 6.4:

P.1 In each drawing, every open configuration contains at least one mountain and one pocket.

P.2 The debt of any open configuration is at most 5π.

Also note that open configurations $\mathcal{C}_\iota$, $\mathcal{C}_j$ and $\mathcal{C}_k$ that did not occur in Section 6.2 will be introduced here. We will consider different cases based on the degree of v_i in the following three subsections. To this end, we mainly consider open configurations $\mathcal{C}_\ell$ and $\mathcal{C}_r$ that include the leftmost and rightmost arc e_ℓ and e_r covered by v_i, respectively. Note that e_ℓ and e_r might not be part of an open configuration, then $\mathcal{C}_\ell = \emptyset$ or $\mathcal{C}_r = \emptyset$.

Insertion of Vertices with Degree two

Case 1: $deg_{G_i}(v_i) = 2$. Here, vertex v_i can interfere only with one open configuration $\mathcal{C} = \mathcal{C}_r = \mathcal{C}_\ell$ since it covers only one arc. If v_i covers at least one pocket (in $\overrightarrow{\Gamma}_i$ or $\overleftarrow{\Gamma}_i$), inserting v_i costs at most $1 + 2\pi$ for both $\overrightarrow{\Gamma}_i$ and $\overleftarrow{\Gamma}_i$. Since $d(\mathcal{C}) \leq 5\pi$, the total cost is at most $1 + 7\pi$ which is at most α as long as $\pi \leq 1/8$.

If v_i covers a mountain in both drawings, $\mathcal{C}$ can only be one of the types $\mathcal{C}_g$, $\mathcal{C}_h$ and $\mathcal{C}_\iota$; see Fig. 6.4. If $\mathcal{C}$ is of type $\mathcal{C}_g$ or $\mathcal{C}_h$, the insertion of v_i creates the open configuration $\mathcal{C}_j$ or $\mathcal{C}_k$, respectively; see Fig. 6.4j or 6.4k. Both of these open

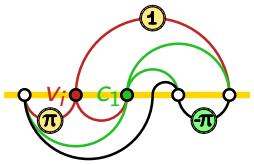


Figure 6.6: Case 2a.1: Drawing when v_i covers a configuration of type $\mathcal{C}_f$.

configurations have a cost $4+3\pi$ for two vertices and hence a debt of 5π. On the other hand, if $\mathcal{C}$ is of type $\mathcal{C}_\iota$, we use the drawings in Fig. 6.5 for the forward and reverse drawing if v_i is inserted on the left mountain; the drawings for the case where v_i is inserted on the right mountain are symmetric. Note that we completely redraw the forward drawing of $\mathcal{C}_\iota$. The costs for inserting v_i and vertices c_1, c_2 are $2+3\pi$ in the forward drawing and $3+2\pi$ in the reverse drawing, achieving a total cost of $5+5\pi$ which is at most 3α if $\pi \leq 1/8$.

Insertion of Vertices with Degree three or four

We treat the cases $deg_{G_i}(v_i) = 3$ and $deg_{G_i}(v_i) = 4$ simultaneously as very often we will apply the same argument. Figures that refer to both cases show the case $deg_{G_i}(v_i) = 4$; the removal of the leftmost vertex yields the corresponding drawing for the case $deg_{G_i}(v_i) = 3$. We distinguish two cases based on whether v_i covers one or more edges of open configuration $\mathcal{C}_r$.

Case 2a: v_i covers at least the two leftmost edges of $\mathcal{C}_r$.

We treat the case where v_i covers at least the two rightmost edge of $\mathcal{C}_\ell$ symmetrically by exchanging the role of forward and reverse drawing in the analysis.

Case 2a.1: $deg_{G_i}(v_i) = 3$. Due to the degree of v_i, it holds that $\mathcal{C}_r = \mathcal{C}_\ell$. Recall that every open configuration contains both a pocket and a mountain on C_{i-1}; cf. Property P.1. More precisely, unless $\mathcal{C}_r = \mathcal{C}_f$, v_i is inserted above a PM pattern in forward or above a MP pattern in reverse drawing. Inserting v_i above such a pattern costs 0 credits while the reverse drawing costs at most 1 credit by Lemma 6.1. Since by Property P.2 $d(\mathcal{C}_r) \leq 5\pi$, the total cost for inserting v_i is at most $1+5\pi$ which is at most α for $\pi \leq 1/6$.

If $\mathcal{C}_r = \mathcal{C}_f$, we redraw $\mathcal{C}_r$ by pushing down the mountain covered by c_1. Then, v_i and c_1 can be placed above the new biarc; see Fig. 6.6. Then, edges incident to c_1 do not need to be assigned credits while we pay for a pocket and a mountain incident to v_i. Since we cover a pocket and reclaim its π credits, we achieve a cost of 1 for both vertices in the forward drawing. As $\mathcal{C}_f$ is symmetric in the reverse drawing, we achieve a total cost of 2 which is less than the 2α credits introduced with v_i and c_1 as long as $\pi \leq 1$.

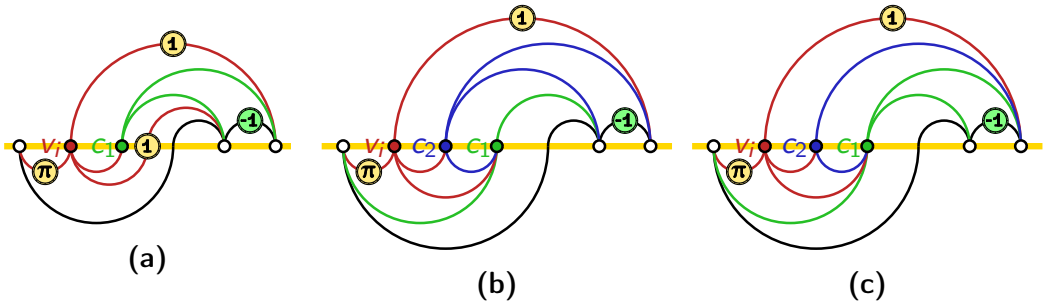


Figure 6.7: Case 2a.2: Drawings when v_i covers a configuration of types (a) $\mathcal{C}_a$, $\mathcal{C}_b$, $\mathcal{C}_c$, $\mathcal{C}_d$ or $\mathcal{C}_g$ (the illustration shows the case where a configuration of type $\mathcal{C}_c$ is covered while e_ℓ is a mountain), (b) $\mathcal{C}_\iota$, and (c) $\mathcal{C}_j$.

Case 2a.2: $deg_{G_i}(v_i) = 4$ and $\mathcal{C}_r$ is completely covered by v_i. We consider the forward drawing $\overrightarrow{\Gamma}_i$ and show that it is cheap enough so that we can even use the default rules discussed before for $\overleftarrow{\Gamma}_i$.

If $\mathcal{C}_r \in \{\mathcal{C}_a, \mathcal{C}_b, \mathcal{C}_c, \mathcal{C}_d, \mathcal{C}_g\}$, we redraw $\mathcal{C}_r$ by placing both v_i and c_1 of $\mathcal{C}_r$ above e_ℓ which we push down if needed; see Fig. 6.7a. While this may create a biarc incident to v_i, we cover all mountains incident to c_1 and hence do not need to allocate credits to them. In addition, we have to pay one credit for mountain (v_i, r_i) and π credits for pocket (ℓ_i, v_i) while we cover at least one mountain covered by c_1. As a result the cost is at most $1 + \pi$ for both vertices in the forward drawing. In the reverse drawing, we pay at most $1 + \pi$ for c_1 by Lemmas 6.1 and 6.2 while v_i is inserted above a pocket by Property P.1, yielding a cost of $1 - \pi$ for inserting v_i into $\overleftarrow{\Gamma}_i$. Hence, we pay at most $3 + 6\pi$ for both vertices, which is at most 2α if $\pi \leq 1/8$.

Next, consider the case where $\mathcal{C}_r \in \{\mathcal{C}_e, \mathcal{C}_f, \mathcal{C}_h\}$. If $\mathcal{C}_r \in \{\mathcal{C}_f, \mathcal{C}_h\}$, v_i is inserted into pocket e_r which costs at most $1 - \pi$ by Lemma 6.1. If $\mathcal{C}_r = \mathcal{C}_e$, we redraw $\mathcal{C}_r$ by putting c_1 in the right pocket it covers instead of the left one; achieving again a cost of $1 - \pi$ for the insertion of v_i. In each case, we observe that the left endpoint of the left mountain of $\mathcal{C}_r$ becomes covered by v_i, hence, it cannot become a biarc in subsequent iterations and we do not need to charge it. Therefore, the forward drawing costs at most $1 - \pi$. In the reverse drawing, inserting c_1 can cost at most $1 + \pi$ by Lemmas 6.1 and 6.2 while v_i is inserted above a pocket, yielding a cost of $1 - \pi$ for inserting v_i. The total cost for inserting both vertices in both drawings then is $3 - \pi$, achieving a cost of at most $3 + 4\pi$ for the insertion of v_i and c_1 and $d(\mathcal{C}_\ell)$. This is less then 2α for $\pi \leq 1/6$.

Finally, consider the case where $\mathcal{C}_r \in \{\mathcal{C}_\iota, \mathcal{C}_j, \mathcal{C}_k\}$. Note that $\mathcal{C}_\ell = \mathcal{C}_r$. If $\mathcal{C}_r \in \{\mathcal{C}_\iota, \mathcal{C}_j\}$, we use the default drawing of c_1 and place both v_i and c_2 into pocket e_ℓ; see Figs. 6.7b and 6.7c, respectively. Since v_i covers both c_1 and c_2 whose incident edges are all proper arc, we only have to assign 1 credit to mountain (v_i, r_i) and π credits to pocket (ℓ_i, v_i). In addition, we can retrieve one credit from

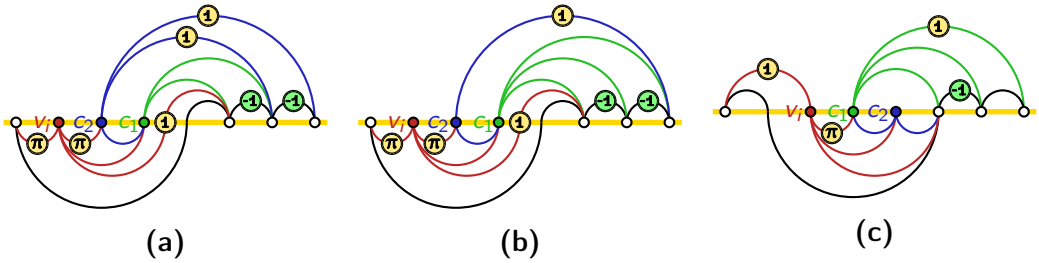


Figure 6.8: Case 2a.3: Drawings when v_i covers two edges of configuration $\mathcal{C}_r$ of type (a) $\mathcal{C}_\iota$, (b) $\mathcal{C}_j$, and (c) $\mathcal{C}_k$ (reverse drawing).

a mountain below $\mathcal{C}_r$. Hence, the forward drawing costs at most π for all three vertices. By Lemmas 6.1 and 6.2, the reverse drawing costs at most $3 + 3\pi$ for all three vertices. We achieve a total cost of $3 + 4\pi$ for both orientation which is at most 3α if $\pi \leq 3/7$. If $\mathcal{C}_r = \mathcal{C}_k$ we apply the same argument for the reverse drawing due to the symmetry of $\mathcal{C}_k$ and $\mathcal{C}_j$.

Case 2a.3: $deg_{G_i}(v_i) = 4$ and v_i covers the two leftmost edges of $\mathcal{C}_r$ but does not cover $\mathcal{C}_r$ completely. Then, $\mathcal{C}_r \in \{\mathcal{C}_\iota, \mathcal{C}_j, \mathcal{C}_k\}$. If $\mathcal{C}_r \in \{\mathcal{C}_\iota, \mathcal{C}_j\}$, we place v_i and vertices c_1 and c_2 of $\mathcal{C}_r$ on edge e_ℓ which we push down if it is a mountain; see Fig. 6.8a and 6.8b, respectively. We have to pay 2π credits for two pockets incident to v_i, possibly one credit for a biarc incident to v_i and up to two credits for mountains incident to c_2, while c_1 is covered without pushing down any of its incident mountains. Since we also cover two mountains below $\mathcal{C}_r$, we achieve a total cost of $1 + 2\pi$ for the forward drawing of the three vertices. By Lemmas 6.1 and 6.2, the reverse drawing can cost at most $3 + 3\pi$ while by Property P.2, $d(\mathcal{C}_\ell) \leq 5\pi$. The total cost for all three vertices therefore is at most $4 + 10\pi$ which is less than 3α if $\pi \leq 2/13$.

If $\mathcal{C}_r = \mathcal{C}_k$, we instead consider the reverse drawing. Again, we place v_i, c_1 and c_2 on top of e_ℓ, which we push down if needed; see Fig. 6.8c. Then, we have to pay at most one credit for edge (ℓ_i, v_i), one credit for mountain (c_1, r_i) and π credits for pocket (v_i, c_1). All remaining edges do not need to carry credits. Moreover, we cover the right endpoint of a mountain covered by c_1, whose credit we can reallocate. As a result, the cost of the reverse drawing here is at most $1 + \pi$. Together with the forward drawing of cost at most $3 + 3\pi$ and $d(\mathcal{C}_\ell) \leq 5\pi$, we achieve a cost of $4 + 9\pi$ for all three vertices which is less than 3α if $\pi \leq 1/6$.

Case 2b: v_i covers only the leftmost edge of $\mathcal{C}_r$.

We assume w.l.o.g. that $d(\mathcal{C}_r) \geq d(\mathcal{C}_\ell)$ since otherwise we can apply a symmetric argument by exchanging the roles of $\overrightarrow{\Gamma}_i$ and $\overleftarrow{\Gamma}_i$. In particular, this is the case if $\mathcal{C}_r = \emptyset$ and v_i covers the rightmost edge of $\mathcal{C}_\ell \neq \emptyset$. Since we already discussed Case 2a, we may assume that only edge e_ℓ belongs to $\mathcal{C}_\ell$. In the cases below, we

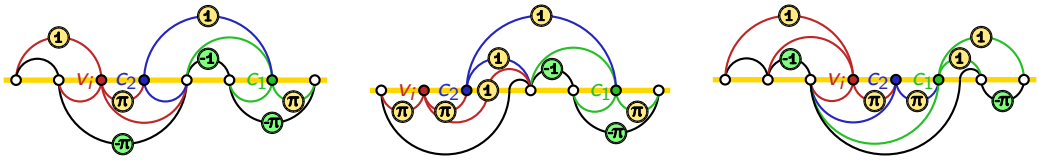


Figure 6.9: Case 2b.1: Forward drawings when v_i covers the leftmost edge of configuration $\mathcal{C}_r$ of type $\mathcal{C}_k$. (a) The edge e^* left of $\mathcal{C}_r$ is a pocket. (b) Edge e^* is a mountain and $deg_{G_i}(v_i) = 3$. (c) Edge e^* is a mountain and $deg_{G_i}(v_i) = 4$.

will pay close attention to the edge e^* that is the covered edge left of $\mathcal{C}_r$, i.e., $e^* = e_\ell$ if $deg_{G_i}(v_i) = 3$. We distinguish several subcases based on the type of $\mathcal{C}_r$, starting with the open configuration types that incur the largest debts.

Case 2b.1: $\mathcal{C}_r = \mathcal{C}_k$. First consider the forward drawing. If e^* is drawn as a pocket, we put both v_i and c_2 above e^* and c_1 into the pocket it covers; see Fig. 6.9a. Then, we need to assign 2π credits to the two pockets (v_i, c_2) and the one incident to c_1 and 1 credit to the mountain (c_2, c_1) and at most one credit to (ℓ_i, v_i). We also cover one mountain and two pockets, hence, the cost for inserting v_i, c_1 and c_2 in the forward drawing is at most 1 in this case.

If e^* is a mountain and $deg_{G_i}(v_i) = 3$, we push down e^* and place v_i, c_1 and c_2 above e^*; see Fig. 6.9b. We have to pay 1 credit for the biarc incident to v_i, 2 credits for the mountains incident to c_2 and 3π for three created pockets, while we can reallocate $1 + \pi$ credits from covered edges. Hence, the cost for inserting v_i, c_1 and c_2 in the forward drawing is at most $2 + 2\pi$ in this case.

If e^* is a mountain and $deg_{G_i}(v_i) = 4$, we push down the mountain covered by $\mathcal{C}_r$ and place v_i, c_1 and c_2 above; see Fig. 6.9c. Then, we pay 3 credits for mountain (v_i, ℓ_i) and the two mountains incident to c_1 and 2π for two pockets incident to c_2. In addition, we reclaim $1 + \pi$ credits from covering e^* and the pocket covered by c_1. Hence, the cost for inserting v_i, c_1 and c_2 in the forward drawing is at most $2 + \pi$ in this case. So, in either case, inserting v_i, c_1 and c_2 in the forward drawing costs at most $2 + 2\pi$.

In the reverse drawing, we place all of v_i, c_1 and c_2 above e^* which we push down if necessary; see Fig. 6.10. We have to pay at most 3 credits for newly created mountains and 2π credits for newly created pockets. In addition, we reclaim at least $1 + \pi$ credits from covered edges. Hence, the reverse drawing can be realized for a cost of at most $2 + \pi$.

Since the forward drawing costs at most $2 + 2\pi$ while the reverse drawing costs at most $2 + \pi$ while $d(\mathcal{C}_\ell) \leq 5\pi$ by Property P.2, the total cost for vertices v_i, c_1 and c_2 is at most $4 + 8\pi$ which is at most 3α for $\pi \leq 2/11$.

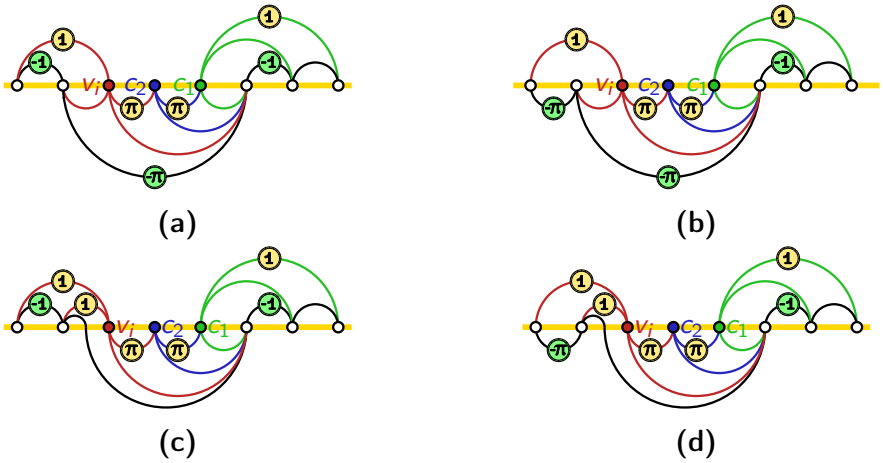


Figure 6.10: Case 2b.1: Reverse drawings when v_i covers the leftmost edge of configuration $\mathcal{C}_r$ of type $\mathcal{C}_k$.

Case 2b.2: $\mathcal{C}_r = \mathcal{C}_j$. Observe that c_2 of $\mathcal{C}_r$ is not incident to v_i. Since $\mathcal{C}_r$ is an open configuration, c_2 did not receive new neighbors after being inserted. Therefore, we can adjust the canonical ordering, so that we insert c_2 directly after v_i; in particular c_2 then does not cover any open configuration while when inserting v_i, $\mathcal{C}_r = \mathcal{C}_g$, which we will discuss below. Note that since we have already discussed open configurations $\mathcal{C}_j$ and $\mathcal{C}_k$, in the following we may assume that $d(\mathcal{C}_\ell) \leq 4\pi$.

Case 2b.3: $\mathcal{C}_r = \mathcal{C}_\iota$. If $deg_{G_i}(v_i) = 4$, we apply the same technique as in Case 2b.2 and insert c_2 of $\mathcal{C}_r$ after v_i. Then, $\mathcal{C}_r = \mathcal{C}_c$ which will be discussed below while c_2 is not covering any open configuration.

On the other hand, if $deg_{G_i}(v_i) = 3$, this cannot be done because v_i is symmetric to c_2 regarding c_1 and this exchange in the canonical ordering would give rise to a new open configuration of type $\mathcal{C}_\iota$. Then, we place v_i above e_ℓ using the default rules. If e_ℓ is a pocket, inserting v_i costs 0 credits, while we can reallocate the credit of the mountain covered by $\mathcal{C}_r$. Hence, the forward drawing costs -1 credit. Since by Lemmas 6.1 and 6.2 the reverse drawing costs at most $1 + \pi$ and $d(\mathcal{C}_r) + d(\mathcal{C}_\ell) \leq 8\pi$, the total cost for inserting v_i is at most 9π which is at most α for $\pi \leq 1/4$.

If e_ℓ is a mountain, we pay for one mountain and one pocket in the forward drawing, while we still cover the mountain covered by $\mathcal{C}_r$. Hence, inserting v_i costs π in the forward drawing. Then, we consider the reverse drawing. If e_ℓ is a mountain in the reverse drawing, $\mathcal{C}_\ell \in \{\emptyset, \mathcal{C}_g, \mathcal{C}_\iota\}$. If $\mathcal{C}_\ell = \emptyset$, the total cost for inserting v_i and paying $d(\mathcal{C}_r)$ is $1 + 6\pi$ which is at most α for $\pi \leq 1/7$. If $\mathcal{C}_\ell \in \{\mathcal{C}_g, \mathcal{C}_\iota\}$, the drawing is symmetric to the one in the forward drawing and costs at most π. Hence, the total cost including $d(\mathcal{C}_r)$ and $d(\mathcal{C}_\ell)$ is at most 10π which is at most α if $\pi \leq 2/11$.

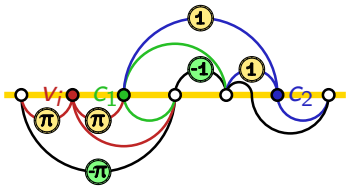


Figure 6.11: Case 2b.3: Non-default reverse drawing when $deg(v_i) = 3$ and $\mathcal{C}_r = \mathcal{C}_\iota$.

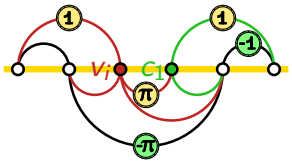


Figure 6.12: Case 2b.4: Forward drawing for the case where e^* is a pocket. The figure illustrates the case where $\mathcal{C}_r = \mathcal{C}_c$.

Finally, if e_ℓ is a pocket in the reverse drawing, we put v_i and c_1 of $\mathcal{C}_r$ in e_ℓ and push down the rightmost mountain covered by c_2 of $\mathcal{C}_r$ to place c_2 there; see Fig. 6.11. We have to pay for two mountains incident to c_2 and two pockets incident to v_i while we reclaim $1 + \pi$ credits from covered edges. The total cost for inserting v_i, c_1 and c_2 in the reverse drawing is $1 + 2\pi$ while we pay at most $3 + 3\pi$ in the forward drawing by Lemmas 6.1 and 6.2. Because $d(\mathcal{C}_\ell) \leq 4\pi$, we can insert all vertices for a total cost of $4 + 9\pi$ which is less than 3α for $\pi \leq 1/6$. Since we now also handled $\mathcal{C}_\iota$, we may assume in the following that $d(\mathcal{C}_\ell) \leq 3\pi$.

Case 2b.4: $\mathcal{C}_r \in \{\mathcal{C}_a, \mathcal{C}_b, \mathcal{C}_c, \mathcal{C}_d, \mathcal{C}_g\}$. If e^* is a pocket, we put v_i and c_1 of $\mathcal{C}_r$ into e^*; see Fig. 6.12. Then we move the π credits from e^* to pocket (v_i, c_1) and put a credit on all created mountains. While this may require several credits, note that for each mountain incident to c_1 and a vertex v_m to the right of c_1 we also cover a mountain with right endpoint v_m. Therefore, we can insert v_i and c_1 for a cost of at most 1 in the forward drawing. By Lemmas 6.1 and 6.2, insertion into the reverse drawing costs at most $2 + 2\pi$. Since $d(\mathcal{C}_\ell) \leq 3\pi$, we insert v_i and c_1 for a total cost of $3 + 5\pi$ which is at most 2α as long as $\pi \leq 1/7$.

If e^* is a mountain and $deg_{G_i}(v_i) = 4$, we use the default drawings for both drawings. In the forward drawing, we observe that e_r is a pocket while e^* is covered and contributes $1 - \pi$ additional free credits compared to what is considered in Lemma 6.1. Hence, we insert v_i for cost 0 into $\overrightarrow{\Gamma}_i$. In the reverse drawing, we obtain a cost of at most $1 - \pi$ by Lemmas 6.1 and 6.2 and we conclude that the total cost including $d(\mathcal{C}_\ell) \leq 3\pi$ is at most $1 + 2\pi$ which is at most α if $\pi \leq 1/3$.

If e^* is a mountain and $deg_{G_i}(v_i) = 3$, we use the default drawing in the forward drawing for c_1 and v_i. By Lemmas 6.1 and 6.2 this costs at most $2 + \pi$ for both vertices. We then consider the reverse drawing. If e^* is a pocket in the

Figure 6.13: Case 2b.4: Reverse drawings where $deg_{G_i}(v_i) = 3$ and e^* is a (a) pocket, or (b) mountain.

reverse drawing, we place both v_i and c_1 of $\mathcal{C}_r$ into it; see Fig. 6.13a. We pay 2π credits for the two pockets incident to v_i and one credit for the mountain between c_1 and its rightmost neighbor. Since we can reallocate the π credits from e^*, this costs at most $1 + \pi$.

If e^* is a mountain in reverse drawing, we distinguish based on the type of $\mathcal{C}_r$. If $\mathcal{C}_r \in \{\mathcal{C}_a, \mathcal{C}_b\}$, we push down e^* to place both v_i and c_1 above; see Fig. 6.13b. Then, we have to pay $2 + \pi$ for two mountains and a pocket while we can reclaim one credits from one of the mountains covered by c_1 achieving again a cost of $1+\pi$ for inserting both v_i and c_1 in the reverse drawing. If $\mathcal{C}_r \in \{\mathcal{C}_d, \mathcal{C}_g\}$, we use the default drawings for both vertices v_i and c_1 which costs at most 2 by Lemmas 6.1 and 6.2. Note that in the process we cover mountain e_ℓ , hence, we pay at most 1 credit for inserting both vertices in the reverse drawing.

Either way, if e^* is a mountain in the reverse drawing and $\mathcal{C}_r \neq \mathcal{C}_c$, inserting v_i and c_1 into the reverse drawing can be done for at most $1 + \pi$ while the cost of the insertion into the forward drawing is at most $2 + \pi$. Since $d(\mathcal{C}_\ell) \leq 3\pi$, we pay at most $3 + 5\pi$ for v_i and c_1 which is at most 2α if $\pi \leq 1/7$. Finally, if $\mathcal{C}_r = \mathcal{C}_c$, we create a new open configuration of type $\mathcal{C}_\iota$.

Case 2b.5: $\mathcal{C}_r \in \{\mathcal{C}_f, \mathcal{C}_h\}$. We use the default drawing for v_i in the forward drawing which costs at most $1 + \pi$ by Lemmas 6.1 and 6.2. As a result, we do not push down e_r. Since there is another mountain below e_r, we can reallocate its credit which reduces the cost for inserting v_i to at most π. Since by Lemmas 6.1 and 6.2 the insertion of v_i into the reverse drawing costs at most $1 + \pi$ and since $d(\mathcal{C}_\ell) \leq 3\pi$, the total cost for inserting v_i is at most $1 + 7\pi$ which is less than α for $\pi \leq 1/8$. Note that, by symmetry, we discussed all cases where $\mathcal{C}_\ell \notin \{\emptyset, \mathcal{C}_d\}$ since $\mathcal{C}_\ell = \mathcal{C}_d$ behaves in the same way as $\mathcal{C}_r = \mathcal{C}_e$. Hence, we can now assume that $d(\mathcal{C}_\ell) \leq \pi$.

Case 2b.6: $\mathcal{C}_r = \mathcal{C}_e$. We use the default drawings for c_1 of $\mathcal{C}_r$ and v_i in the forward drawing which costs at most $1 - \pi$ by Lemma 6.1 since both cover a pocket. Thus, the forward drawing costs at most $2 - 2\pi$ for both c_1 and v_i.

If e^* is a pocket in the reverse drawing, we put both v_i and c_1 in it; see Figs. 6.14a and 6.14b. If e^* is a mountain and $deg_{G_i}(v_i) = 3$ or e_ℓ is a mountain,

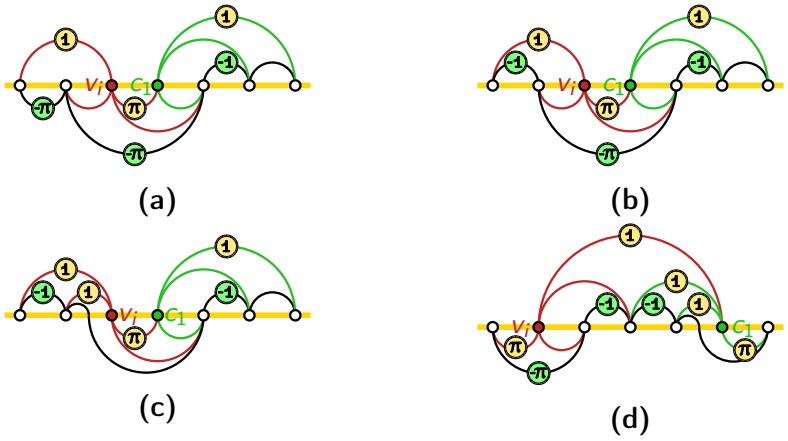


Figure 6.14: Case 2b.6: Reverse drawings when v_i covers the leftmost edge of configuration $\mathcal{C}_r$ of type $\mathcal{C}_e$.

we push down e^* and put both v_i and c_1 in it; see Figs. 6.14c. In each of these cases, we have 1 credit to pay for mountain (ℓ_i, v_i) and if e^* is a mountain π for pocket (v_i, c_1). If $deg_{G_i}(v_i) = 4$, e^* is a mountain and e_ℓ is a pocket, we use the default drawing for both v_i and c_i, for a cost of $1 + \pi$; see Fig. 6.14d.

Since $d(\mathcal{C}_\ell) \leq \pi$, it follows that inserting v_i and c_1 into both drawings costs at most 3 which is at most 2α if $\pi \leq 1/2$.

Cases 2a, and 2b cover all possible subcases. In either case, we have shown that if $\pi \leq 1/8$, we can either insert v_i at a cost of at most α (while a covered open configuration may be redrawn) or create a new open configuration or change the canonical ordering locally to reduce to some simpler case.

Insertion of Vertices with Degree at least five

If $deg_{G_i}(v_i) \geq 5$, it may cover several open configurations, some completely (i.e. all arcs of the open configuration) and some partially. The general idea is that completely covered open configurations pay for themselves due to Property P.1: The left endpoint of their mountain on the outer face becomes covered which frees an allocated credit that can be used to pay their debt. However, the partially covered open configurations must be more carefully considered. We show in the following how to insert v_i in the forward drawing for a cost of at most $1 - \pi$ which includes the coverage of debt $d(\mathcal{C}_r)$ and the debts of all fully covered open configurations except $\mathcal{C}_\ell$ as long as $\pi \leq 1/8$. By a symmetric argument, v_i can be inserted into the reverse drawing for at most $1 - \pi$ including paying the debt $d(\mathcal{C}_\ell)$. This then implies that v_i can be inserted for a cost of $2 - 2\pi$.

Case 3a: v_i covers only mountains.

By Property P.1, v_i cannot fully cover any open configuration. In addition, by Lemma 6.2, inserting v_i costs at most 0 credits. Since $d(\mathcal{C}_r) \leq 5\pi$ by Property P.2 introduced at the beginning of this section, we can insert v_i into $\overrightarrow{\Gamma}_i$ for $1 - \pi$ as long as $\pi \leq 1/6$.

Case 3b: e_r is a pocket.

Here, v_i is inserted into e_r. We pay 1 credit for mountain (ℓ_i, v_i) and π for pocket (v_i, r_i) which we can reclaim for e_r. In addition, we can reallocate the credits of all remaining covered edges, i.e., 1 for every mountain and π for every pocket. The least amount of credits that we can obtain that way is achieved, when e_ℓ is a mountain (whose credit we cannot reallocate) while all remaining edges are pockets. We achieve the following bound for the insertion cost of v_i:

$$c \leq 1 - (deg_{G_i}(v_i) - 3)\pi \leq 1 - 2\pi \tag{6.1}$$

This bound even holds if one of the covered mountains m is part of an open configuration $\mathcal{C}_m$. Namely, if $m = e_\ell$, we account for $d(\mathcal{C}_m)$ in the reverse drawing. Otherwise, m is allocated a full credit in contrast to the π credits used in the calculation of Eq. (6.1). Since by Property P.2 $d(\mathcal{C}_m) \leq 5\pi$, we can pay the debt $d(\mathcal{C}_m)$ without violating Eq. (6.1) if $\pi \leq 1/6$.

If we fully cover $\mathcal{C}_r$, the mountain of $\mathcal{C}_r$ is not e_r and we are done. The same is true if $d(\mathcal{C}_r) = \pi$. Otherwise, we consider four subcases based on the edge e'_r occuring to left of e_r along C_{i-1}.

Case 3b.1: e'_r is a mountain and not part of an open configuration. We gain at least $1 - \pi$ more credits compared to Eq. (6.1) from covering e'_r. Those are sufficient to pay $d(\mathcal{C}_r)$ if $\pi \leq 1/6$.

Case 3b.2: e'_r is a mountain and part of the open configuration $\mathcal{C}_r$. Then $\mathcal{C}_r = \mathcal{C}_\iota$. Moreover, v_i covers the left endpoint q of $\mathcal{C}_r$. Vertex q is the left endpoint of two mountains. We gain at least one more credit compared to Eq. (6.1) from covering q which we can use to cover debt $d(\mathcal{C}_r)$.

Case 3b.3: e'_r is a mountain and part of an open configuration $\mathcal{C}'_r \neq \mathcal{C}_r$. Then, e'_r is the rightmost edge of $\mathcal{C}'_r$. First, consider the left endpoint q of e'_r. If $\mathcal{C}'_r \notin \{\mathcal{C}_c, \mathcal{C}_e, \mathcal{C}_\iota, \mathcal{C}_j\}$, vertex q is the left endpoint of two mountains. Then, when covering q, we can reallocate an extra credit compared to Eq. (6.1) that we can use to cover $d(\mathcal{C}_r)$.

If $\mathcal{C}'_r = \mathcal{C}_c$, we completely cover $\mathcal{C}'_r$. Then, we reposition vertex c_1 of $\mathcal{C}'_r$ into e_r which allows to redraw the edge covered by c_1 as a mountain instead of as a

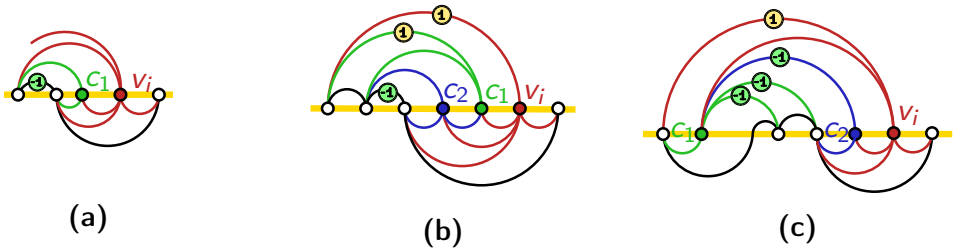


Figure 6.15: Case 3b.3: Redrawing of open configuration $\mathcal{C}'_r$ that involves edge e'_r. (a) $\mathcal{C}'_r$ is of type $\mathcal{C}_c$. (b) $\mathcal{C}'_r$ is of type $\mathcal{C}_\iota$. (c) $\mathcal{C}'_r$ is of type $\mathcal{C}_j$.

biarc; see Fig. 6.15a. We save an additional $1-\pi$ credits compared to Eq. (6.1) that is sufficient to cover $d(\mathcal{C}_r)$ if $\pi \leq 1/4$.

Similarly, if $\mathcal{C}'_r = \mathcal{C}_e$, we redraw $\mathcal{C}'_r$ such that c_1 is put into the right pocket covered by c_1 instead. Then, e'_r becomes a pocket and we proceed with the discussion of Case 3b.4 below.

If $\mathcal{C}'_r = \mathcal{C}_\iota$, we redraw $\mathcal{C}'_r$ such that both c_1 and c_2 appear in pocket e_r; see Fig. 6.15b. This way, we only have to charge one mountain incident to c_1 with the credits introduced with c_1 and c_2 while we also cover another mountain. Hence, the 2α credits introduced with c_1 and c_2 can be reallocated to pay $d(\mathcal{C}_r)$.

Finally, if $\mathcal{C}'_r = \mathcal{C}_j$, we move vertex c_2 of $\mathcal{C}'_r$ on top of edge e_r. This way, we can redraw the two biarcs incident to c_1 as mountains; see Fig. 6.15c. Since c_1 gets covered, we can reclaim the two credits of the mountains to pay $\mathcal{C}_r$.

Case 3b.4: e'_r is a pocket (which may be part of an open configuration). This case can only occur if $\mathcal{C}_r \in \{\mathcal{C}_a, \mathcal{C}_b, \mathcal{C}_c, \mathcal{C}_g, \mathcal{C}_j, \mathcal{C}_k\}$. If $\mathcal{C}_r \in \{\mathcal{C}_j, \mathcal{C}_k\}$, there are two pockets belonging to $\mathcal{C}_r$ and e_r can be either one of them. First consider the case, where e_r is the left pocket of $\mathcal{C}_r$. We move the vertices c_1 and possibly c_2 of $\mathcal{C}_r$ into pocket e_r right of v_i. This allows to redraw the biarc below c_1 and c_2 as a mountain; see Figs. 6.16a and 6.16b for illustrations where $\mathcal{C}_r$ is of type $\mathcal{C}_j$ and $\mathcal{C}_k$, respectively. Since we cover the left endpoint of the newly created mountain, we gain a surplus of at least $1-\pi$ credits compared to Eq. (6.1) which suffices to pay for $d(\mathcal{C}_r)$ if $\pi \leq 1/6$.

Hence, it remains to discuss the case where e'_r is the leftmost pocket of $\mathcal{C}_r \in \{\mathcal{C}_j, \mathcal{C}_k\}$. If $\mathcal{C}_r = \mathcal{C}_j$, we move c_2 and v_i into pocket e'_r. Then, we can redraw both biarcs incident to c_1 as mountains; see Fig. 6.16c. Since c_1 is covered by v_i, we can use the credits of these two mountains to cover debt $d(\mathcal{C}_r)$.

If $\mathcal{C}_r = \mathcal{C}_k$, consider the edge e''_r left of e'_r on C_{i-1}. If e''_r is a pocket, we redraw $\mathcal{C}_r$ as follows: We move c_1 into the pocket covered by $\mathcal{C}_r$ and c_2 into pocket e''_r. Then, we insert v_i into e''_r to the left of c_2; see Fig. 6.16d. Compared to the default drawing, we can reallocate two credits; one of the edge below $\mathcal{C}_r$ that can be redrawn as a biarc and one credit that is allocated to a mountain incident to c_1. This suffices to pay debt $d(\mathcal{C}_r)$ if $\pi \leq 2/5$.

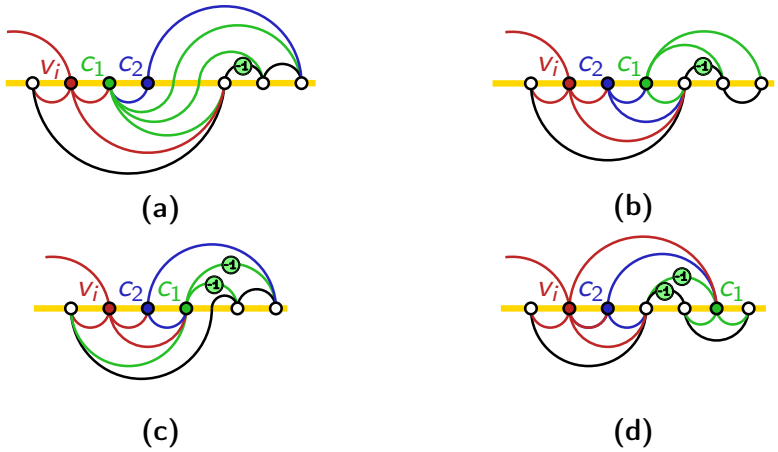


Figure 6.16: Case 3b.4: Redrawing of open configuration $\mathcal{C}_r$ if edge e'_r is a pocket. (a)–(b) e'_r is not part of $\mathcal{C}_r$. (c)–(d) e'_r is part of $\mathcal{C}_r$.

If e''_r is a mountain, we distinguish two cases. If e''_r is part of no open configuration or of $\mathcal{C}_\ell$, we gain an additional $1-\pi$ credits from covering its left endpoint that are not accounted for in Eq. (6.1). These suffice to pay for $d(\mathcal{C}_r)$ if $\pi \leq 1/6$. Otherwise, e''_r is part of an open configuration $\mathcal{C}''_r \neq \mathcal{C}_\ell$ which cannot be of types $\mathcal{C}_f$ or $\mathcal{C}_h$ since e''_r is the rightmost arc of $\mathcal{C}''_r$. Since $\mathcal{C}''_r \neq \mathcal{C}_\ell$ we cover it completely. Unless $\mathcal{C}''_r \in \{\mathcal{C}_c, \mathcal{C}_e, \mathcal{C}_j\}$, there exists another mountain below $\mathcal{C}''_r$ whose left endpoint becomes covered by v_i. The reclaimed credit can be used to pay for $d(\mathcal{C}_r)$ if $\pi \leq 1/5$. If $\mathcal{C}''_r = \mathcal{C}_e$, we redraw $\mathcal{C}''_r$ by moving c_1 into the right pocket it covers. This does not change the costs, but e''_r becomes a pocket and we use the argument above. Next, if $\mathcal{C}''_r = \mathcal{C}_c$, we move c_1 into e'_r which allows the biarc below $\mathcal{C}''_r$ to be redrawn as a mountain. We obtain another $1-\pi$ credits that can be reallocated which suffices to pay for $d(\mathcal{C}_r)$ if $\pi \leq 1/6$. Finally, the case where $\mathcal{C}''_r = \mathcal{C}_j$ can be resolved similarly by moving c_1 and c_2 into pocket e'_r which even yields $2-\pi$ credits to redistribute.

Case 3c: v_i covers a pocket and e_r is a mountain.

We apply Lemma 6.1 to insert v_i for a cost of $1-\pi$ by placing v_i into the rightmost covered pocket p. As in Case 3b, all fully covered open configurations left of p except for $\mathcal{C}_\ell$ can pay for their own debt since by Property P.1 they have a mountain on the outer face whose left endpoint becomes covered. Moreover since all covered edges right of p are mountains, by Property P.1 there are at most two open configurations whose debts where not covered yet, namely, $\mathcal{C}_r$ and possibly $\mathcal{C}_p$ to which p belongs to. If $\mathcal{C}_r \neq \mathcal{C}_p$, the rightmost edge of $\mathcal{C}_r$ is mountain e_r and it can only be of types $\mathcal{C}_f$, $\mathcal{C}_h$ and $\mathcal{C}_\iota$. For each of those, there exists another mountain m below e_r whose credit can be reallocated when inserting v_i. This extra credit can be used to pay for $\mathcal{C}_r$ if $\pi \leq 1/5$.

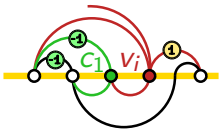


Figure 6.17: Case 3c: Redrawing of open configuration $\mathcal{C}_p = \mathcal{C}_c$.

Hence, it remains to discuss open configuration $\mathcal{C}_p$. If $\mathcal{C}_p = \mathcal{C}_\ell$, we will pay its debt in the reverse drawing. Hence, assume now that $\mathcal{C}_p$ is completely covered. If $\mathcal{C}_p \in \{\mathcal{C}_f, \mathcal{C}_h, \mathcal{C}_\iota\}$, there is a mountain of $\mathcal{C}_p$ to the left of p that is entirely covered. Since we do not account for it in Lemma 6.1, we obtain $1 - \pi$ extra credits that can be used to pay $d(\mathcal{C}_p)$ if $\pi \leq 1/6$.

If $\mathcal{C}_p \in \{\mathcal{C}_a, \mathcal{C}_b, \mathcal{C}_d, \mathcal{C}_g, \mathcal{C}_k\}$, there is at least one mountain in $\mathcal{C}_p$ which is not on the outer face but whose left endpoint is covered by v_i. We can use this extra credit to pay $d(\mathcal{C}_p)$ if $\pi \leq 1/5$. For $\mathcal{C}_p = \mathcal{C}_j$, we can use the rearrangement discussed in Case 3b.4 with the difference that v_i also covers c_2; see Fig. 6.16c.

If $\mathcal{C}_p = \mathcal{C}_e$, we redraw $\mathcal{C}_p$ such that both c_1 and v_i are inserted into the right pocket covered by c_1. This redrawing does not alter the cost of the drawing, however, we can now use the credit on the mountain incident to c_1 to pay the debt of $\mathcal{C}_p$ if $\pi \leq 1/5$.

Finally, consider the case where $\mathcal{C}_p = \mathcal{C}_c$. If $\mathcal{C}_p = \mathcal{C}_r$, we use the argument from Case 3b. Otherwise, consider the mountain m to the right of p on C_{i-1}. If m is e_r, $\mathcal{C}_r$ can only be of types $\mathcal{C}_f$, $\mathcal{C}_h$ and $\mathcal{C}_\iota$ and $d(\mathcal{C}_r) \leq 4\pi$. In addition, there is another mountain below e_r, whose credit can be used to pay for $d(\mathcal{C}_r) + d(\mathcal{C}_p) \leq 7\pi$ as long as $\pi \leq 1/7$. Otherwise, m is not part of any open configuration. Then we push down m and place c_1 of $\mathcal{C}_p$ and v_i on top; see Fig. 6.17. While m becomes a biarc, v_i has one less mountain to its right, hence, the cost for inserting v_i stays the same. On the other hand, we can redraw the edge covered by c_1 as a mountain, which gives us another credit, that we can use to pay $d(\mathcal{C}_p)$ as long as $\pi \leq 1/3$.

Cases 3a, 3b and 3c cover all situations. In either case, we have shown that if $\pi \leq 1/8$, we can insert v_i in $\overrightarrow{\Gamma_i}$ while paying the debts for all partially covered open configurations except for $\mathcal{C}_\ell$ for a cost of at most $1 - \pi$ as claimed.

6.4 Proof of Theorem 5.1

In this section, we now prove the main theorem of this chapter:

Theorem 6.1. *Every planar graph admits a plane down-up monotone arc diagram with at most $\lfloor 15/16n - 5/2 \rfloor$ biarcs that can be computed in linear time.*

Proof. In Sections 6.2 and 6.3, we showed that vertices can be inserted at a cost of at most α credits in both drawings while maintaining Invariants I.1 to I.6 in both drawings if $\pi \leq 1/8$. Therefore, we choose $\pi = 1/8$ to obtain the largest savings.

Here, we still have to discuss the initial drawings of G_3, the open configurations present in G_n and the running time of our algorithm.

The initial drawings $\overrightarrow{\Gamma}_3$ and $\overleftarrow{\Gamma}_3$ are identical, in particular, we have v_1 as leftmost and v_2 as rightmost vertex while all edges are realized as pockets. Therefore, we assign π credits each to pockets (v_1, v_3) and (v_3, v_2) in both drawings. Hence, after assigning credits to the initial drawing, we have $3\alpha - 4\pi = 6 - 7\pi > 5$ credits left from the credits arriving with vertices v_1, v_2 and v_3. Moreover, these credits are still remaining after inserting v_n. This is the case as in G_n only one open configuration can be left as the upper envelope consists only of edges (v_1, v_n) and (v_n, v_2). Moreover, by Property P.1, if those two edges belong to an open configuration, one is realized as a mountain that carries an entire credit while the debt is at most 5π by Property P.2. We conclude that we create at most $\alpha n - 5 = 15/8n - 5$ biarcs in both drawings or $\lfloor 15/16n - 5/2 \rfloor$ in the drawing with less biarcs.

Finally, consider the runtime. A canonical ordering can be computed in linear time. In addition, when inserting a vertex, we only consider the vertices it covers and its leftmost and rightmost neighbor while we only redraw such parts of the drawing. This results in a linear runtime. □

6.5 Description of the SAT Formulation

In this section, we describe a SAT formulation that can check whether a planar input graph G can be realized as an arc diagram with a prespecified number of biarcs κ. Our formulation is based on a SAT formulation that can decide whether G admits a k-page stack layout [39]. Most notably, we did the following adjustments:

(i) We restricted the SAT formulation to only two pages.

(ii) For every edge e_i, we introduce a dummy vertex d_{e_i} that represents the spine crossing of a potential biarc and variables β_i^j for $1 \leq j \leq \kappa$. We can enforce d_{e_i} to be located in between the endpoints of e_i to obtain a monotone arc diagram.

(iii) An edge $e = (u, v)$ needs to be assigned to a page only if it is not a biarc. Otherwise, its two *half-edges* (u, d_e) and (d_e, v) must be assigned to different pages.

(iv) We check intersections between pairs of edges and/or half-edges only if both are assigned to the same page.

The resulting formulae are of size $\Theta(n^3)$ for a graph on n vertices. Therefore, an actual implementation can only be used for small values of n when running on a standard workplace machine. We experimentally verified the following:

Observation 6.1. *No Kleetope based on a triangulation with up to* 14 *vertices requires more biarcs in its down-up monotone arc diagrams compared to its general arc diagrams even if the outer face is arbitrarily prescribed.*

We generated the input triangulations with the program `plantri` [49]. Also, we tested several other triangulations without more concrete results. In the following, we describe the clauses of the SAT formulation. We start with required clauses and then describe optional clauses. We describe our clauses also using boolean operators $\oplus$ (exclusive or), $\Rightarrow$ (implication) and $\Leftrightarrow$ (equivalence) which can be easily transformed into conjunctive normal form.

Required Clauses

Vertex Ordering. We first enumerate vertices and edges arbitrarily from 1 to n and 1 to m, respectively. The dummy vertex d_{e_k} of edge e_k is represented as vertex v_{n+k}. We introduce a variable $\sigma_{i,j}$ for each pair of vertices v_i, v_j which we interpret as follows: If $\sigma_{i,j} = \top$, we assume that v_i appears before v_j on the spine, otherwise, v_j appears before v_i. In order to obtain a total ordering of the vertices on the spine, we simply must ensure antisymmetry, i.e.,

$$(\sigma_{i,j} \oplus \sigma_{j,i}) \quad \forall 1 \leq i < j \leq n+m$$

and transivity, i.e.,

$$((\sigma_{i,j} \wedge \sigma_{j,k}) \Rightarrow \sigma_{i,k}) \wedge ((\sigma_{k,j} \wedge \sigma_{j,i}) \Rightarrow \sigma_{k,i}) \quad \forall 1 \leq i < j < k \leq n+m.$$

Reflexitivity does not need to be ensured as the position of a vertex to itself will not be checked.

Bounding the Number of Biarcs. As mentioned before, we have variables β_i^j for $1 \leq j \leq \kappa$ for edge e_i. In particular, edge e_i will be interpreted as a biarc if and only $\beta_i^j = \top$ for at least one $j \in \{1, \ldots, \kappa\}$. If $\beta_i^j = \top$, we enforce that $\beta_k^j = \bot$ for all edges $e_k \neq e_i$ to ensure that there are at most κ biarcs as follows:

$$\bigwedge_{1 \leq i < k \leq m} (\neg\beta_i^j \vee \neg\beta_k^j) \quad \forall 1 \leq j \leq k.$$

Note that this creates $\Theta(n^3)$ clauses.

Page Assignment. Each edge and half-edge that is present in the resulting arc diagram must be assigned to one page. In particular, for edge $e_k = (v_i, v_j)$ with $i < j$ there are two half-edges $h_{k,1} = (v_i, v_{n+k})$ and $h_{k,2} = (v_j, v_{n+k})$. For the sake of brevity, we also define $h_{k,0} = e_k$. For edge e_k we introduce six page variables $\phi_{k,i,j}$ with $0 \leq i \leq 2$ and $1 \leq j \leq 2$. We interpret $\phi_{k,i,j} = \top$ such that half-edge

$h_{k,i}$ is drawn on page j where w.l.o.g. page 1 is the bottom and page 2 the top page. In order to obtain a valid page assignment, an edge e_k must be labelled as a biarc or it must be drawn on one of the two pages:

$$\left(\bigvee_{1\leq j<\kappa}(\beta_k^j)\vee\phi_{k,0,1}\vee\phi_{k,0,2}\right)\quad\forall 1\leq k\leq m.$$

In addition, if e_k happens to be a biarc, we require that its two half-edges are drawn on two different pages:

$$\begin{aligned}(\beta_k^j\Rightarrow(\phi_{k,1,1}\vee\phi_{k,2,1}))&\quad\forall 1\leq k\leq m\quad\forall 1\leq j<\kappa\\(\beta_k^j\Rightarrow(\phi_{k,1,2}\vee\phi_{k,2,2}))&\quad\forall 1\leq k\leq m\quad\forall 1\leq j<\kappa\\(\neg\phi_{k,1,1}\vee\neg\phi_{k,1,2})&\quad\forall 1\leq k\leq m\\(\neg\phi_{k,2,1}\vee\neg\phi_{k,2,2})&\quad\forall 1\leq k\leq m.\end{aligned}$$

Planarity. An intersection between a pair of edges and/or half-edges can only occur if they are drawn on the same halfplane. Hence, we first introduce a variable $\chi_{k,k',i,i'}$ for $1\leq k\leq k'$ and $0\leq i,i'\leq 2$. We interpret $\chi_{k,k',i,i'}=\top$ such that half-edges $h_{k,i}$ and $h_{k',i'}$ are located on the same page and thus can possibly conflict. This can be formulated as follows:

$$\bigwedge_{1\leq j\leq 2}(\phi_{k,i,j}\wedge\phi_{k',i',j}\Rightarrow\chi_{k,k',i,i'})\quad\forall 0\leq i,i'\leq 2\quad\forall 1\leq k<k'\leq m$$

After computing $\chi_{k,k',i,i'}$, it is easy to avoid intersections. For this, let $h_{k,i}=\left(v_{s(k,i)},v_{t(k,i)}\right)$ and $h_{k',i'}=\left(v_{s(k',i')},v_{t(k',i')}\right)$. We avoid intersections as follows: if $h_{k,i}$ and $h_{k',i'}$ would intersect if they are on the same page, we require that both are not on the same page, i.e., $\chi_{k,k',i,i'}=\bot$:

$$\begin{aligned}(\sigma_{s(k,i),s(k',i')}\wedge\sigma_{s(k',i'),t(k,i)}\wedge\sigma_{t(k,i),t(k',i')}&\Rightarrow\neg\chi_{k,k',i,i'})\\(\neg\sigma_{s(k,i),s(k',i')}\wedge\neg\sigma_{s(k',i'),t(k,i)}\wedge\neg\sigma_{t(k,i),t(k',i')}&\Rightarrow\neg\chi_{k,k',i,i'})\\(\sigma_{t(k,i),s(k',i')}\wedge\sigma_{s(k',i'),s(k,i)}\wedge\sigma_{s(k,i),t(k',i')}&\Rightarrow\neg\chi_{k,k',i,i'})\\(\neg\sigma_{t(k,i),s(k',i')}\wedge\neg\sigma_{s(k',i'),s(k,i)}\wedge\neg\sigma_{s(k,i),t(k',i')}&\Rightarrow\neg\chi_{k,k',i,i'})\\(\sigma_{s(k,i),s(k',i')}\wedge\sigma_{s(k',i'),t(k,i)}\wedge\sigma_{t(k,i),t(k',i')}&\Rightarrow\neg\chi_{k,k',i,i'})\\(\neg\sigma_{s(k,i),t(k',i')}\wedge\neg\sigma_{t(k',i'),t(k,i)}\wedge\neg\sigma_{t(k,i),s(k',i')}&\Rightarrow\neg\chi_{k,k',i,i'})\\(\sigma_{t(k,i),t(k',i')}\wedge\sigma_{t(k',i'),s(k,i)}\wedge\sigma_{s(k,i),s(k',i')}&\Rightarrow\neg\chi_{k,k',i,i'})\\(\neg\sigma_{t(k,i),t(k',i')}\wedge\neg\sigma_{t(k',i'),s(k,i)}\wedge\neg\sigma_{s(k,i),s(k',i')}&\Rightarrow\neg\chi_{k,k',i,i'})\end{aligned}$$

Optional Clauses

Monotone Biarcs. The monotonicity of biarcs can be achieved by requiring dummy vertex $d_{e_k} = v_{n+k}$ of edge $e_k = (v_i, v_j)$ to be located in between v_i and v_j:

$$((\sigma_{i,n+k} \wedge \sigma_{n+k,j}) \vee (\sigma_{j,n+k} \wedge \sigma_{n+k,i})) \quad \forall e_k = (v_i, v_j) \in E$$

Conversely, if we omit such clauses, we can compute a non-monotone arc diagram instead.

Down-Up Biarcs. Consider edge $e_k = (v_i, v_j)$. Recall that half-edge $h_{k,1}$ is incident to v_i while $h_{k,2}$ is incident to v_j. Since $\sigma_{i,j}$ defines whether v_i is to the left of v_j, we can enforce the left half-edge to be located on the bottom page if e_k is drawn as a biarc as follows:

$$\bigwedge_{1 \le \ell \le \kappa} ((\beta_k^\ell \wedge \sigma_{i,j}) \Rightarrow \phi_{k,1,1}) \wedge ((\beta_k^\ell \wedge \sigma_{j,i}) \Rightarrow \phi_{k,2,1}) \quad \forall e_k = (v_i, v_j) \in E$$

Prescription of an Outer Face for a Triangulation. If the input graph G is a triangulation, consider a face $f_0 = (v_i, v_k, v_k)$. First, we ensure that the leftmost and the rightmost vertex of face f_0 are also the leftmost and the rightmost vertex of G, respectively:

$$\begin{aligned}
((\sigma_{i,j} \wedge \sigma_{i,k}) \Rightarrow \sigma_{i,\ell}) \wedge ((\sigma_{j,i} \wedge \sigma_{k,i}) \Rightarrow \sigma_{\ell,i}) \quad & \forall v_\ell \in V \setminus \{v_i, v_j, v_k\} \\
((\sigma_{j,i} \wedge \sigma_{j,k}) \Rightarrow \sigma_{j,\ell}) \wedge ((\sigma_{i,j} \wedge \sigma_{k,j}) \Rightarrow \sigma_{\ell,j}) \quad & \forall v_\ell \in V \setminus \{v_i, v_j, v_k\} \\
((\sigma_{k,j} \wedge \sigma_{k,i}) \Rightarrow \sigma_{k,\ell}) \wedge ((\sigma_{j,k} \wedge \sigma_{i,k}) \Rightarrow \sigma_{\ell,k}) \quad & \forall v_\ell \in V \setminus \{v_i, v_j, v_k\}
\end{aligned}$$

Moreover, the edges of f_0 must be restricted such that all remaining vertices can be placed only inside f_0. To this end, we assign the edge between the leftmost and the rightmost vertex on page 1 while the other two edges of f_0 are required to be located on page 2. We define $e_{ij} = (v_i, v_j)$, $e_{ik} = (v_i, v_k)$ and $e_{jk} = (v_j, v_k)$ and obtain the following constraints:

$$\begin{aligned}
((\sigma_{i,j} \Leftrightarrow \sigma_{j,k}) \Rightarrow (\phi_{ik,0,1} \wedge \phi_{ij,0,2} \wedge \phi_{jk,0,2})) \\
((\sigma_{j,k} \Leftrightarrow \sigma_{k,i}) \Rightarrow (\phi_{ij,0,1} \wedge \phi_{jk,0,2} \wedge \phi_{ik,0,2})) \\
((\sigma_{k,i} \Leftrightarrow \sigma_{i,j}) \Rightarrow (\phi_{jk,0,1} \wedge \phi_{ij,0,2} \wedge \phi_{ik,0,2})).
\end{aligned}$$

Faster Computation Time for Kleetopes. For a Kleetope G based on a triangulation T, we can speed up the computation based on the following observation: An arc diagram of G can be obtained from an arc diagram Γ of T if for every

face f in T, the new dummy vertex v_f has access to the spine inside f. This is the case if any edge bounding f is a biarc or if f contains both an edge assigned to page 1 and an edge assigned to page 2. Let e_i , e_j and e_k be the edges bounding f. Then:

$$\bigwedge_{1\leq \ell \leq 2} \left(\phi_{i,0,\ell} \vee \phi_{j,0,\ell} \vee \phi_{k,0,\ell} \vee \bigvee_{1\leq \iota \leq \kappa} (\beta_i^\iota \vee \beta_j^\iota \vee \beta_k^\iota) \right).$$

Part III

Beyond Planar Drawings

Density and Area Bounds for Polyline RAC Drawings

Chapter 7

Density Bounds for RAC Drawings with one Bend per Edge

One of the main research directions of graph drawing beyond planarity is bounding the maximum edge density of the class of graphs realizable in a certain beyond-planar drawing style. This parameter gives a first intuition of the complexity of graphs that may be drawn. In the literature, this type of problem is also known as *Turán type* and has been studied for many different graph classes beyond planarity; see e.g. [2, 3, 4, 5, 6, 13, 37, 56, 97, 118, 136, 137, 144].

The maximum edge density of polyline RAC drawings has been already investigated in the first paper on RAC drawings [71] where it was shown that (i) the density of RAC_0 graphs is at most $4n - 10$ which is a tight bound, while, (ii) the density of RAC_1 and RAC_2 graphs is subquadratic, whereas, (iii) every graph admits a RAC_3 drawing. Subsequently, Arikushi et al. [19] proved linear upper bounds for the edge density for RAC_1 and RAC_2 graphs, in particular, RAC_1 graphs have at most $6.5n - 13$ edges while RAC_2 graphs have less than $74.2n$ edges. Also, they showed that there are infinitely many RAC_1 and RAC_2 graphs with $4.5n - \mathcal{O}(\sqrt{n})$ edges and and $7.83n - \mathcal{O}(\sqrt{n})$ edges, respectively.

In this chapter[1], we close the gap between upper and lower bound of the edge density of RAC_1 graphs up to an additive constant. Namely, we prove that their maximum edge density is $5.5n - 11$ while we also demonstrate that there are infinitely many RAC_1 graphs with $5.5n - 72$ edges. Additionally, we investigate the class of simple RAC_1 graphs, that is, the class of graphs admitting simple RAC_1 drawings. Here, we show that the maximum edge density is at most $5.4n - 10.8$ while we give a lower bound of $5n - 10$. To the best of our knowledge, this result makes RAC_1 graphs the second class of beyond planar graphs besides quasiplanar graphs [5] for which a difference in the maximum edge density between the general

[1] The results of this chapter also appeared in [12].

setting and the setting in which simplicity of the drawings is required is known.

Our upper bound results are derived by a refinement of the charging technique used by Arikushi et al. [19] which we describe in Section 7.1. Then, we describe our adjustment and prove our new upper bounds in Section 7.2. Finally, we present our lower bound constructions in Section 7.3.

7.1 Overview of the Charging Scheme

In this section we present the most important notation and technical details of the charging scheme used by Arikushi et al. [19] for bounding the maximum edge density of RAC$_1$ graphs which we will make use of in Section 7.2. Let $G = (V, E)$ be a simple RAC$_1$ graph and let Γ be a RAC$_1$ drawing of G with the minimum number of intersections. We partition E into sets E_0 and E_1 such that E_0 contains all edges that are intersection-free in Γ while E_1 contains all edges that have at least one intersection in Γ. Further, we denote by G_0 and G_1 the subgraphs induced by E_0 and E_1, respectively. When we mention G_0 and G_1 in this chapter, we will implicitly assume that they are derived from a RAC$_1$ drawing Γ. Note that since G_0 is a planar graph, it follows that $|E_0| \leq 3n - 6$.

In order to bound the number of edges belonging to E_1, consider the planarization $G_1' = (V_1', E_1')$ of G_1 and the set of faces F_1' of G_1'. In the following, we denote by $deg(v)$ the degree of v in G_1' and by $s(f)$ the *size* of face $f \in F_1'$, which is the number of edges encountered in a walk along the boundary of f. Note that $s(f)$ counts each edge on the boundary of f as often as it appears in such a walk (i.e. once or twice). We assign an initial charge $ch(v) = deg(v) - 4$ to each vertex $v \in V_1'$ and an initial charge $ch(f) = s(f) - 4$ to each face of $f \in F_1'$. Using Euler's formula, the sum of initial charge is

$$\sum_{v \in V_1'} (deg(v) - 4) + \sum_{f \in F_1'} (s(f) - 4) = 2|E_1'| - 4|V_1'| + 2|E_1'| - 4|F_1'| = -8.$$

Afterwards, the charges are redistributed in two phases such that (i) the total charge of G_1' is maintained, and, (ii) all faces have non-negative charge.

In the first discharging phase, we redistribute charges from the endpoints of edges in E_1 with a bend to the faces for which the bend forms a convex corner. Let $(u, v) \in E_1$ be an edge with a bend b and let f be the face for which b is convex. Then, we move $1/2$ charge from u to f and $1/2$ charge from v to f; see Fig. 7.1a. It is shown in [19] that every face of size less than four is incident to at least one convex bend and therefore receives at least one unit of charge in this phase. Moreover, it is also shown that *lenses*, that is, faces of size two, are incident to a second bend. Thus, the only faces which still have a negative charge after the first discharging phase are the *reflex* lenses, that is, faces of size two that have one convex bend and one reflex bend. More precisely, the charge for a reflex

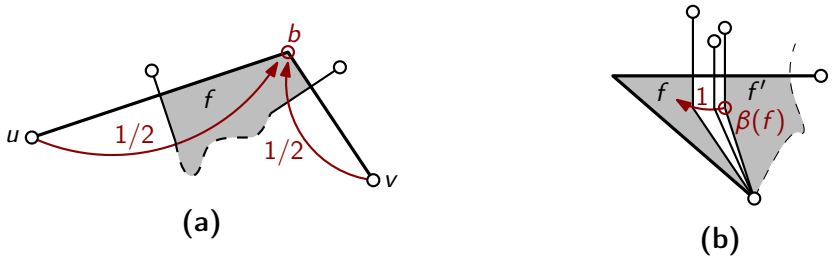


Figure 7.1: Redistribution of charges in the two discharging phases in [19].

lens f is now $ch'(f) = -1$. On the other hand, for faces $f \in F_1'$ that are not a reflex lens, the new charge is $ch'(f) \geq 0$. Since for every edge incident to a vertex v we redistributed $1/2$ of its charges, the charge of vertex $v \in V$ after the first discharging phase is $ch'(v) \geq 1/2 deg(v) - 4$ while for $v \in V_1' \setminus V$ (that is, the dummy vertices) it is still $ch'(v) = ch(v) = 0$.

For the second discharging phase, Arikushi et al. [19] prove that there is an injective mapping β between reflex lenses and convex bends incident to faces in F_1' that have size at least four. Let f be a reflex lens and f' be the face containing $\beta(f)$. Since $s(f') \geq 4$, it holds that $ch(f') \geq 0$. On the other hand, in the first discharging phase f' was assigned at least one additional charge because it is incident to the convex bend $\beta(f)$. We now move this charge to the reflex lens f; see Fig. 7.1b. As a result, the new charge of f is $ch''(f) = 0$. Clearly, for all faces $f \in F_1'$ that are not reflex lenses, it holds that $ch''(f) \geq ch'(f) \geq 0$ if they are not involved in the discharging phase or $ch''(f) \geq ch(f) \geq 0$ if they are one of the faces of length at least four involved in one of the dischargings. Since for all vertices $ch''(v) = ch'(v)$, it follows

$$|E_1| - 4n = \sum_{v \in V} (1/2 deg(v) - 4) \leq \sum_{v \in V_1'} ch''(v) + \sum_{f \in F_1'} ch''(f) = -8, \qquad (7.1)$$

which implies that

$$|E_1| \leq 4n - 8. \qquad (7.2)$$

This immediately gives that $|E_0| + |E_1| \leq 7n - 14$. However, this bound can be clearly improved as G_0 is a triangulation if $|E_0| = 3n - 6$ which then implies that $E_1 = \emptyset$ since a planar triangle cannot contain any edge of E_1.

In order to give a better bound, Arikushi et al. [19] next consider how many edges can belong to E_1 if G_0 can be obtained from a triangulation by the removal of k edges. Let F_0 denote the set of faces of G_0 and let $d(f)$ denote the *degree* of a face $f \in F_0$, that is, the number of distinct vertices on the boundary of f. By applying Eq. (7.2) to the subgraphs of G_1' inside each of the faces of F_0, we obtain the following new bound for the number of edges in E_1:

$$|E_1| \leq \sum_{\substack{f \in F_0 \\ d(f) > 3}} (4d(f) - 8) \qquad (7.3)$$

Arikushi et al. [19] then prove that the right-hand side of Eq. (7.3) is upper-bounded by $8k$. Since we will apply a similar refined argument in Section 7.2, we provide their proof here. The argument is inductive on the number of edges k that where removed from a triangulation to obtain G_0. For $k = 0$, all faces have degree three and the right-hand side of Eq. (7.3) is $0 = 8k$. Assume now that if G_0 is obtained from a triangulation by removing k edges, the right-hand side of Eq. (7.3) is at most $8k$. Then, consider the case where G_0 is obtained by removing $k+1$ edges. Clearly, G_0 can be obtained from a graph G_0' by removing one edge e while G_0' is obtained from a triangulation by removing k edges. For G_0' the right-hand side of Eq. (7.3) is at most $8k$. We consider four cases depending of e.

C.1 If e is a bridge of a face[2] in G_0', removing e yields a new face in G_0 of the same size, hence the right-hand side of Eq. (7.3) does not change and remains at most $8k < 8(k+1)$.

C.2 If e is adjacent to two triangles of G_0', in G_0 both triangles become a face of degree four. Hence, the right-hand side of Eq. (7.3) increases by at most 8 to at most $8k + 8 = 8(k+1)$.

C.3 If e is adjacent to a triangle and a face f of degree $d(f) > 3$ in G_0', these two faces create a face of degree at most $d(f)+1$ in G_0. The right-hand side of Eq. (7.3) decreases by $4d(f) - 8$ since f is not part of G_0 anymore but increases by $4(d(f)+1) - 8 = 4d(f) - 4$, yielding a net increase of 4. Thus the right-hand side of Eq. (7.3) is now at most $8k + 4 < 8(k+1)$.

C.4 If e is adjacent to two faces f_1 and f_2 of degrees $d(f_1) > 3$ and $d(f_2) > 3$ in G_0', respectively, the removal of e creates a face of degree $d(f_1)+d(f_2)-2$ in G_0. The right-hand side of Eq. (7.3) decreases by $4(d(f_1)+d(f_2)) - 16$ since f_1 and f_2 are not part of G_0 anymore but increases by $4(d(f_1)+d(f_2)-2)-8 = 4(d(f_1)+d(f_2)) - 16$, yielding a net increase of 0. Thus the right-hand side of Eq. (7.3) remains at most $8k < 8(k+1)$.

Based on this observation, Arikushi et al. [19] derive two new bounds for the number of edges in G. Since G_0 is obtained from a triangulation by removing k edges, it follows that

$$|E| = |E_0| + |E_1| \leq (3n - 6 - k) + (4n - 8) = 7n - 14 - k \qquad (7.4)$$

On the other hand, since the right-hand side of Eq. (7.3) is at most $8k$, we also obtain

$$|E| = |E_0| + |E_1| \leq (3n - 6 - 8) + (8k) = 3n - 6 + 7k \qquad (7.5)$$

[2]A bridge of a face is an edge whose removal separates a facial walk of the face into two.

The minimum of the two bounds (7.4) and (7.5) is maximized when $k = n/2 - 1$ providing the upper bound

$$|E| \leq 6.5n - 13.$$

It is noteworthy, that Arikushi et al. [19] point out that their upper bound is an overestimation. In particular, they state that investigating faces of small degree can be worthwhile. This can be easily observed in Case C.2 which is the only one where the right-hand side of Eq. (7.3) increases by eight – the resulting face however is a quadrangle and can only contain two intersecting edges in contrast to the eight edges in Eq. (7.2). We investigate this behavior in the next section.

7.2 Upper Bound Results

In this section, we improve the analysis of the charging scheme discussed in Section 7.1 which results in a better upper bound for the maximum edge density of RAC_1 graphs. Mainly, we improve two aspects of the analysis. First, we analyze the structure of faces in the planarization G_1' more carefully. Second, we treat faces that can be triangulated with few edges separately; this will also allow us to find a clear difference between RAC_1 and simple RAC_1 graphs.

We assume w.l.o.g. that G is connected and has at least 5 vertices. Consider a face f of G_0. Recall that we denote by $d(f)$ the degree of f, that is, the number of vertices on the boundary of f. Since inside f there can be edges of G_1, f is not necessarily connected. Instead, its boundary is composed of a disjoint set of (not necessarily simple) cycles which we call *facial walks* and isolated vertices; see Fig. 7.2a. In particular, we consider isolated edges as a cycle of length two.

We denote by $\ell(f)$ the *length* of face f, that is the sum of the lengths of all facial walks of f. Note that some edges can be counted twice; for instance, edges (v_1, v_2) and (v_9, v_{10}) are appearing twice in the facial walks of the graph in Fig. 7.2a. Observe that such behavior is caused when a vertex v appears more than once in a facial walk. We denote by $m_f(v)$ the number of occurences of v in facial walks of f minus one, that is, the number of multiple occurences beyond its first occurence. Moreover, for face f, we denote the number of such extra occurences among all its vertices with $m(f)$, that is, $m(f) = \sum_{v \in f} m_f(v)$. In addition, we denote by $b(f)$ the number of biconnected components of all facial walks of f. We do not consider isolated vertices as biconnected components, but every edge that is either a bridge of f or that is an entire connected component is considered a biconnected component. Moreover, the number of isolated vertices of f is denoted by $i(f)$. Since every vertex v except for the $i(f)$ isolated vertices appears $m_f(v)+1$ times in some facial walk of f, we conclude that $\ell(f) = d(f) + m(f) - i(f)$.

Consider the planarization G' of G obtained from drawing Γ. Since the edges E_0 are part of G', every face of G' is contained inside the polygon bounding one of the faces of G_0. We denote by $F'(f)$ the set of faces of G' that is contained

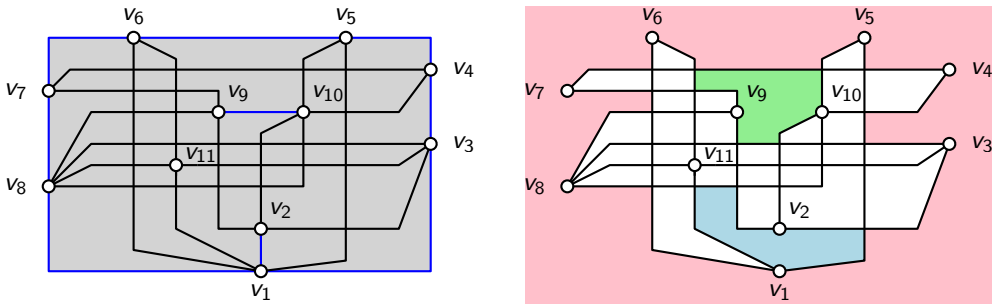


Figure 7.2: (a) A non-simple disconnected face f of G_0 (edges colored blue) that also contains edges of G_1 (colored black). Face f consists of one isolated vertex, namely v_{11}, and two facial walks, namely $w_1 = (v_1, v_2, v_1, v_3, v_4, v_5, v_6, v_7, v_8)$ and $w_2 = (v_9, v_{10})$. Clearly $d(f) = 11$ and $i(f) = 1$. Further, $\ell(f) = 11$ since the sum of lengths of w_1 and w_2 is 11. Because v_1 appears twice in w_1 and separates w_1 into two biconnected components, we also have $m(f) = 1$ and $b(f) = 3$. Face f is good. Note that the removal of edge (v_4, v_7) would result in f not being good anymore because edges (v_5, v_6) and (v_9, v_{10}) would appear in the same face of G'. (b) The three faces of $F_1'(f)$ that surround the biconnected components of f are highlighted. Note that the length of each of these faces is longer than twice the length of the corresponding biconnected components.

inside face f. We call a face f of G_0 *good* if and only if f is a triangle or each face in $F'(f)$ contains at most one planar edge. For instance, the face shown in Fig. 7.2a is good. In the next two lemmas we assume that all faces of G_0 are good. Afterwards we show how to deal with drawings in which some face is not good; in this process we may introduce parallel edges (but no self-loops) in G_0, which are non-homotopic[3] by construction. Such edges may not be drawable with just one bend, however, the geometry of the edges in G_0 does not affect the discharging scheme of Arikushi et al. [19] as discussed in Section 7.1. We first give a better bound on the number of intersected edges inside a face of G_0 by using the more detailed description of faces discussed above.

Lemma 7.1. *Let Γ be a drawing of G such that all faces of G_0 are good. Each face f of G_0 contains at most $2d(f) - 2m(f) + 2i(f) + 4b(f) - 8$ edges of G_1.*

Proof. Consider the graph $G(f)$ which is the subgraph of G which is induced by the interior of f in Γ. Further, let $\Gamma(f)$ be the subdrawing of Γ representing $G(f)$. In addition, let $G_1(f) = (V_1(f), E_1(f))$ be the subgraph of $G_(f)$ induced by the set of edges that intersect in $\Gamma(f)$ and let $G'(f)$ be the planarization of $G_1(f)$.

Consider the set of biconnected components $B(f)$ of f and the set of faces $F_1'(f)$ of $G_1'(f)$ that are inside the polygon bounding f. Because face f is good, in

[3]Two parallel edges are non-homotopic if each region that is bounded by both edges contains at least one vertex.

$G_1'(f)$, the two endpoints u and v of an edge on the boundary of f are connected by paths of length at least two. More precisely, this path contains at least one dummy vertex in between u and v. Hence, every biconnected component $c \in B(f)$ is surrounded by a face $f_c' \in F_1'(f)$ of length $\ell(f_c') \geq 2\ell(c)$ in G_1'; see Fig. 7.2b. In the charging scheme described in Section 7.1, the initial charge of f_c' is $ch(f_c') = \ell(f_c') - 4 \geq 2\ell(c) - 4$. After the second discharging phase, the charge of each face is at least as much as its initial charge, thus, $ch''(f_c') \geq 2\ell(c) - 4$. Note that an isolated vertex is not surrounded by a single face of $G_1'(f)$. We sum up the charge of all faces surrounding biconnected components of f:

$$\begin{aligned} \sum_{c \in B(f)} ch''(f_c') &\geq \sum_{c \in B(f)} (2\ell(c) - 4) = 2(\ell(f) - 4b(f)) \\ &= 2(d(f) + m(f) - i(f)) - 4b(f) \end{aligned}$$

After the second discharging phase, every face has a non-negative charge. In addition, the charges of faces surrounding biconnected components of f is a lower bound for the sum of charges in all faces of $F_1'(f)$. Hence, it holds that

$$\sum_{f' \in F_1'(f)} ch''(f') - \sum_{c \in B(f)} ch''(f_c') \geq 0.$$

This allows us to refine Eq. (7.1) for $G(f)$ as follows:

$$\begin{aligned} |E_1(f)| - 4d(f) &= \sum_{v \in f} (1/2deg(v) - 4) \leq \sum_{v \in f} ch''(v) \\ &\leq \sum_{v \in f} ch''(v) + \sum_{f' \in F_1'(f)} ch''(f') - \sum_{c \in B(f)} ch''(f_c') \\ &\leq -8 - 2(d(f) + m(f) - i(f)) + 4b(f) \end{aligned}$$

which yields the required result. □

Next, similar to the analysis in [19], we consider how many edges are contained in G_1, when the planar subgraph G_0 is obtained from a triangulation T by removing k edges under the assumption that all faces are good. Recall that we assume that G_0 may contain parallel edges as long as they are non-homotopic. Consider a face $f \in F_0$. We denote by $t(f)$ the number of edges needed to triangulate f. Observe that $k = \sum_{f \in F_0} t(f)$. We first consider faces that can be triangulated with few edges. To this end, let $F_0^i = \{f \in F_0 | t(f) = i\}$. First, if $t(f) = 0$, f is a planar triangle and does not contain any edge of G_1; see Fig. 7.3a. If $t(f) = 1$, f can only be a quadrangle or a face bounded by two facial walks of length two; see Figs. 7.3b or 7.3c, respectively. In these two cases, f can contain at most two edges of G_1. Moreover, if $t(f) = 2$, the boundary of f is one of the four possible configurations shown in Figs. 7.3d–7.3g. In these cases, f contains at most five intersecting edges, namely, if f is a pentagon; see Fig. 7.3d. By Lemma 7.1 and

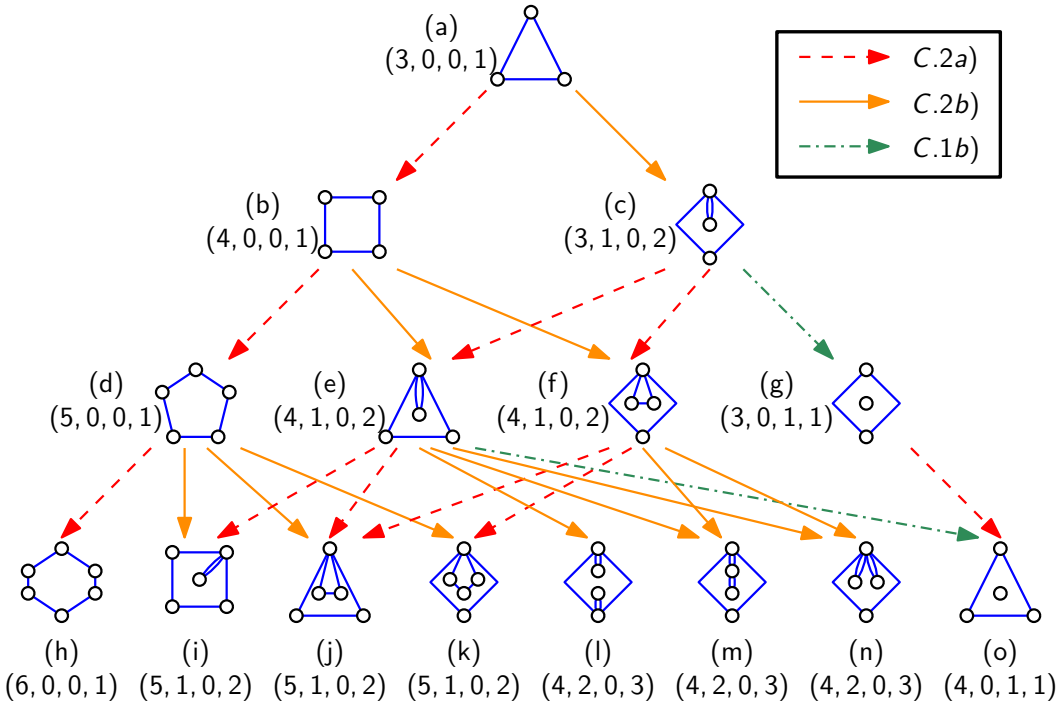


Figure 7.3: All bounded faces that can be triangulated with at most three edges: (a) $t(f) = 0$, (b)–(c) $t(f) = 1$, (d)–(g) $t(f) = 2$, and, (h)–(o) $t(f) = 3$. The caption of each subfigure lists the values $(d(f), m(f), i(f), b(f))$ and the arrows in between two configurations show how faces can be obtained from each other by the removal of edges according to Cases C.1 and C.2. Parallel edges on the boundary of a face indicate that either two non-homotopic parallel edges are present or that the face contains a bridge.

our observations, we conclude that

$$|E_1| \leq 2|F_0^1| + 5|F_0^2| + \sum_{\bigcup_{i=3}^{k} F_0^i} (2d(f) - 2m(f) + 2i(f) + 4b(f) - 8). \qquad (7.6)$$

For simple drawings, we will additionally prove that f contains at most seven edges if $t(f) = 3$; the possible boundaries of such faces are listed in Figs. 7.3h–7.3o. Note that this is clearly true for the faces shown in Figs. 7.3i–7.3o by considering how many edges are missing from the complete graph. In the case where f is a simple hexagon, however, nine edges are missing to make the subgraph complete; see Fig. 7.3h. With this result, we can refine Eq. (7.6) as follows:

$$|E_1| \leq 2|F_0^1| + 5|F_0^2| + 7|F_0^3| + \sum_{\bigcup_{i=4}^{k} F_0^i} (2d(f) - 2m(f) + 2i(f) + 4b(f) - 8). \qquad (7.7)$$

In the following lemma, we will show that overestimations of the right-hand sides of Eqs. (7.6) and (7.7) are upperbounded by $\frac{8}{3}k$ and $\frac{5}{2}k$, respectively. Since those

right-hand sides are upper bounds for $|E_1|$, this implies that $|E_1| \leq \frac{8}{3}k$ for not necessarily simple RAC_1 drawings and that $|E_1| \leq \frac{5}{2}k$ for simple RAC_1 drawings.

Lemma 7.2. *Let G be a RAC_1 graph with a drawing Γ such that G_0 is obtained from a triangulation T by removal of k edges. Then,*

$$\frac{8}{3}|F_0^1| + \frac{16}{3}|F_0^2| + \sum_{\bigcup_{i=3}^{k} F_0^i} (2d(f) - 2m(f) + 2i(f) + 4b(f) - 8) \leq \frac{8}{3}k, \quad (7.8)$$

and, if Γ is simple,

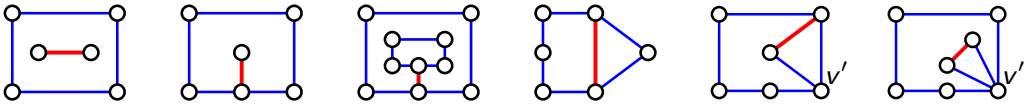
$$\frac{5}{2}|F_0^1| + 5|F_0^2| + \frac{15}{2}|F_0^3| + \sum_{\bigcup_{i=4}^{k} F_0^i} (2d(f) - 2m(f) + 2i(f) + 4b(f) - 8) \leq \frac{5}{2}k. \quad (7.9)$$

Proof. Similar to the corresponding proof of Lemma 5 in [19], we prove the statement by induction on k. However, we assume that we obtain G_0 from T by removing edges in a specific order. Namely, we want to avoid the case, where an edge between two distinct non-triangular faces is removed; see Case C.4 in Section 7.1. This can be guaranteed as follows: Consider the subgraph D of the dual graph of T that is induced by the edges dual to those that we remove from T to obtain G_0. The edges of D are removed in an order in which they appear in a BFS traversal of each connected component of D. Then, each interlevel edge in the BFS traversal corresponds to removing an edge incident to a not yet visited triangular face, while each intralevel edge corresponds to removing a bridge of a face that was created in prior steps. In neither case, we merge two non-triangular faces as unwished.

Let $\tau(G_0)$ and $\tau_S(G_0)$ denote the left-hand sides of Eq. (7.8) and Eq. (7.9), respectively. For the base case of the induction, $k = 0$ and $G_0 = T$. Then, $\tau(G_0) = \tau_S(G_0) = 0$ and Eqs. (7.8) and (7.9) hold. For the induction hypothesis, assume that Eqs. (7.8) and (7.9) hold for $k \geq 0$.

For the induction step, consider a RAC_1 graph G' whose planar subgraph G'_0 is obtained from a triangulation T by removing $k' = k + 1$ edges. Let G_0 be the plane graph obtained by the removal of these k' edges except for the last one, say (u, v) in the ordering according to the BFS traversal discussed above. Recall that the ordering of edge deletions ensures that (u, v) is not separating two distinct non-triangular faces. Hence, the difference between G_0 and G'_0 is that G'_0 contains a face f' with u and v on its boundary, while G_0 contains up to two faces in G_0 with (u, v) on their boundaries; only one of which, say f, can be non-triangular. By the induction hypothesis, for G_0, it holds that $\tau(G_0) \leq \frac{8}{3}k$ and $\tau_S(G_0) \leq \frac{5}{2}k$. We consider the following possible cases:

C.1 Edge (u, v) is a bridge of some face f in G_0. Let f' be the face of G'_0 that results when removing (u, v) from f. Clearly, $t(f') = t(f) + 1$. Since every bridge is a biconnected component, it holds that $b(f') = b(f) - 1$. Since



(a) C.1a **(b)** C.1b **(c)** C.1c **(d)** C.2a **(e)** C.2b **(f)** C.2b

Figure 7.4: Illustration of Cases C.1 and C.2. Edge (u, v) is colored red.

(u, v) is only on the boundary of f, it also holds that $d(f') = d(f)$. Consider the following subcases:

C.1a u and v become isolated vertices in G_0'; see Fig. 7.4a. Clearly, $i(f') = i(f) + 2$. Since neither u nor v appeared twice in facial walks of G_0, it also holds that $m(f') = m(f)$.

C.1b Exactly one of u and v, say u, becomes an isolated vertex in G_0'; see Fig. 7.4b. Clearly, $i(f') = i(f) + 1$. Since v is still connected to a facial walk and since it was connected to u with the biconnected component (u, v), also $m(f') = m(f) - 1$.

C.1c Neither u nor v become an isolated vertex in G_0'; see Fig. 7.4c. Then, $i(f') = i(f)$. Moreover, both u and v were connected to at least two biconnected components of facial walks before, one of which was edge (u, v). Hence $m(f') = m(f) - 2$.

C.2 Removing (u, v) merges a triangular face Δ and an adjacent face f of G_0 creating a face f' of G_0'. Again, $t(f') = t(f) + 1$. We consider two cases:

C.2a Δ and f share only edge (u, v); see Fig. 7.4d. In other words, f' is obtained from f by subdividing an edge along one of its biconnected components once. Hence, $d(f') = d(f) + 1$, $m(f') = m(f)$, $b(f') = b(f)$ and $i(f') = i(f)$.

C.2b Δ and f share at least two edges; see Figs. 7.4e and 7.4f. In other words, f' is obtained from f by splitting a biconnected component of f with length ℓ into two new biconnected components with total boundary length $\ell + 1$. Both of these components are incident to a cut vertex v' which is neither u nor v. Hence, $d(f') = d(f)$, $m(f') = m(f) + 1$, $b(f') = b(f) + 1$ and $i(f') = i(f)$.

In other words, $\tau(G_0')$ is equal to $\tau(G_0)$ plus the contribution of f' to $\tau(G_0')$ minus the contribution of f to $\tau(G_0)$; the analogous fact is true for $\tau_S(G_0')$ and $\tau_S(G_0)$. Next, consider $t(f')$.

First, if $t(f') = 1$, f is a planar triangle and f' is obtained by merging f with another triangle as described in Case C.2. In Case C.2a, f' is a simple quadrangle; see Fig. 7.3b. In Case C.2b, the biconnected facial walk of f is split into two biconnected components of length two; see Fig. 7.3c. Since $f' \in F_0^1$

and $f \in F_0^0$, it follows that $\tau(G_0') = \tau(G_0) + \frac{8}{3} - 0 \leq \frac{8}{3}k + \frac{8}{3} = \frac{8}{3}k'$ and $\tau_S(G_0') = \tau_S(G_0) + \frac{5}{2} - 0 \leq \frac{5}{2}k + \frac{5}{2} = \frac{5}{2}k'$.

Next, consider $t(f') = 2$, that is, f is one of the two configurations in Figs. 7.3b and 7.3c. Using one of the following three operations, we obtain the configurations depicted in Figs. 7.3d–7.3g for f':

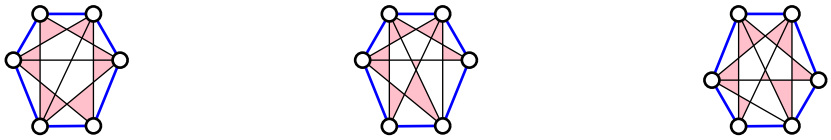
(i) Applying Case C.2a, the length of a biconnected component of a facial walk of f is increased by one; see the red dashed arrows in Fig. 7.3. For instance, one of the two biconnected components of length two in Fig. 7.3c can be subdivided to create the configurations shown in Figs. 7.3e and 7.3f.

(ii) Applying Case C.2b, a biconnected component of a facial walk of f with length $\ell \geq 3$ is split into two biconnected components of total lengths $\ell + 1$; see the solid orange arrows in Fig. 7.3. For instance, the biconnected component of length four in Fig. 7.3b can be split to create the two configurations shown in Figs. 7.3e and 7.3f. Note that both of these configurations can be obtained in two different ways.

(iii) Applying Case C.1b, a bridge that is incident to some biconnected component of a facial walk is removed creating an isolated vertex; see the dash-dotted green arrows in Fig. 7.3. For instance, removing the bridge in the configuration in Fig. 7.3c yields the new configuration shown in Fig. 7.3g.

Since $f' \in F_0^2$ and $f \in F_0^1$, it follows that $\tau(G_0') = \tau(G_0) + \frac{16}{3} - \frac{8}{3} \leq \frac{8}{3}k + \frac{8}{3} = \frac{8}{3}k'$ and $\tau_S(G_0') = \tau_S(G_0) + 5 - \frac{5}{2} \leq \frac{5}{2}k + \frac{5}{2} = \frac{5}{2}k'$.

Next, consider $t(f') = 3$. Starting from the possible configurations for f shown in Figs 7.3d–7.3g, we use the same rules as discussed in the previous case to obtain the configurations shown in Figs. 7.3h–7.3o for face f'. Note that in Fig. 7.3, we also report the values for $d(f')$, $m(f')$, $b(f')$ and $i(f')$. With this information, it is possible to verify that $2d(f') - 2m(f') + 2i(f') + 4b(f') - 8$ for each of the configurations shown in Figs. 7.3h–7.3. Since $f' \in F_0^3$ and $f \in F_0^2$, it follows that $\tau(G_0') = \tau(G_0) + 8 - \frac{16}{3} \leq \frac{8}{3}k + \frac{8}{3} = \frac{8}{3}k'$. For the case of simple drawings, it is also possible to show that f' contains at most seven edges of G_1. Recall that G_1 does not contain parallel edges. In the configurations shown in Figs. 7.3i–7.3o there are only at most five vertices and at least three distinct edges. Since the complete graph K_5 has ten edges, it follows that there are at most seven edges of G_1 in such faces. Hence, it remains to consider the configuration where f' is a simple hexagon; see Fig. 7.3h. In a simple drawing, no two edges of G_1 inside f' share more than one point which can be either an intersection or a common endpoint. Hence, the graph $K_6 - e$, that is, the graph obtained from K_6 by removing an edge e, has only three topological embeddings with the given outer face containing all six vertices; see Fig. 7.5. Note that these topologies are uniquely defined by the sequence of intersections along all edges due to the simplicity of the drawing. In each of these three embeddings, there are ten triangular regions which each are incident to at



(a) (b) (c)

Figure 7.5: All topologocial embeddings of $K_6 - e$ where all vertices are incident to the outer face. Note that each such topology contains ten triangular regions (shaded red) which each are incident to at least two intersections.

least two intersections; see the red-shaded regions in Fig. 7.5. Since all intersections occur at a right angle, each such triangular region must contain a convex bend. However, there are only eight edges with one bend each, a contradiction. Thus, for simple RAC$_1$ drawings, $\tau_S(G_0') = \tau_S(G_0) + 7 - 5 < \tau_S(G_0) + \frac{15}{2} - 5 \leq \frac{5}{2}k + \frac{5}{2} = \frac{5}{2}k'$.

Finally, consider $t(f') \geq 4$. We reconsider Cases 1 (removal of an edge (u, v) that is a bridge of a face f) and 2 (removal of an edge (u, v) that separates a face f and another triangular face Δ).

C.1a Here, u and v become isolated vertices after removing edge (u, v). As shown above, $d(f') = d(f)$, $m(f') = m(f)$, $i(f') = i(f) + 2$ and $b(f') = b(f) - 1$. Because

$$\begin{aligned} & 2d(f') - 2m(f') + 2i(f') + 4b(f') - 8 \\ = \; & 2d(f) - 2m(f) + 2(i(f) + 2) + 4(b(f) - 1) - 8 \\ = \; & 2d(f) - 2m(f) + 2i(f) + 4b(f) - 8, \end{aligned}$$

we conclude that $\tau(G_0') = \tau(G_0) \leq \frac{8}{3}k < \frac{8}{3}k'$ and $\tau_S(G_0') = \tau_S(G_0) \leq \frac{5}{2}k < \frac{5}{2}k'$.

C.1b Here, one of u and v becomes an isolated vertex after removing edge (u, v). As shown above, $d(f') = d(f)$, $m(f') = m(f) - 1$, $i(f') = i(f) + 1$ and $b(f') = b(f) - 1$. Because

$$\begin{aligned} & 2d(f') - 2m(f') + 2i(f') + 4b(f') - 8 \\ = \; & 2d(f) - 2(m(f) - 1) + 2(i(f) + 1) + 4(b(f) - 1) - 8 \\ = \; & 2d(f) - 2m(f) + 2i(f) + 4b(f) - 8, \end{aligned}$$

we conclude that $\tau(G_0') = \tau(G_0) \leq \frac{8}{3}k < \frac{8}{3}k'$ and $\tau_S(G_0') = \tau_S(G_0) \leq \frac{5}{2}k < \frac{5}{2}k'$.

C.1c Here, u and v become part of two disconnected biconnected components when removing (u, v). As shown above, $d(f') = d(f)$, $m(f') = m(f) - 2$,

$i(f') = i(f)$ and $b(f') = b(f) - 1$. Because

$$\begin{aligned} & 2d(f') - 2m(f') + 2i(f') + 4b(f') - 8 \\ = \; & 2d(f) - 2(m(f) - 2) + 2i(f) + 4(b(f) - 1) - 8 \\ = \; & 2d(f) - 2m(f) + 2i(f) + 4b(f) - 8, \end{aligned}$$

we conclude that $\tau(G_0') = \tau(G_0) \leq \frac{8}{3}k < \frac{8}{3}k'$ and $\tau_S(G_0') = \tau_S(G_0) \leq \frac{5}{2}k < \frac{5}{2}k'$.

C.2a Here, faces f and Δ share only vertices u and v. As shown above, $d(f') = d(f) + 1$, $m(f') = m(f)$, $i(f') = i(f)$ and $b(f') = b(f)$. Because

$$\begin{aligned} & 2d(f') - 2m(f') + 2i(f') + 4b(f') - 8 \\ = \; & 2(d(f) + 1) - 2m(f) + 2i(f) + 4b(f) - 8 \\ = \; & 2d(f) - 2m(f) + 2i(f) + 4b(f) - 8 + 2, \end{aligned}$$

we conclude that $\tau(G_0') = \tau(G_0) + 2 \leq \frac{8}{3}k + 2 < \frac{8}{3}k'$ and $\tau_S(G_0') = \tau_S(G_0) + 2 \leq \frac{5}{2}k + 2 < \frac{5}{2}k'$.

C.2b Here, all vertices on the boundary of Δ are also on the boundary of f. As shown above, $d(f') = d(f)$, $m(f') = m(f) + 1$, $i(f') = i(f)$ and $b(f') = b(f) + 1$. Because

$$\begin{aligned} & 2d(f') - 2m(f') + 2i(f') + 4b(f') - 8 \\ = \; & 2d(f) - 2(m(f) + 1) + 2i(f) + 4(b(f) + 1) - 8 \\ = \; & 2d(f) - 2m(f) + 2i(f) + 4b(f) - 8 + 2, \end{aligned}$$

we conclude that $\tau(G_0') = \tau(G_0) + 2 \leq \frac{8}{3}k + 2 < \frac{8}{3}k'$ and $\tau_S(G_0') = \tau_S(G_0) + 2 \leq \frac{5}{2}k + 2 < \frac{5}{2}k'$.

Finally, we consider the case $t(f') = 4$ for simple RAC_1 drawings independently to conclude the proof. Observe that in each of the cases, $\tau_S(G_0') \leq \tau_S(G_0) + 2$. As stated before, for all configurations where $t(f) = 3$, we have that $2d(f) - 2m(f) + 2i(f) + 4b(f) - 8 \leq 8$; see Figs. 7.3h–7.3o. Thus, it follows that $2d(f') - 2m(f') + 2i(f') + 4b(f') - 8 \leq 8 + 2 = 10$. Since $f' \in F_0^4$ and $f \in F_0^3$, it holds that $\tau_S(G_0') = \tau_S(G_0) + 10 - \frac{15}{2} \leq \frac{5}{2}k + \frac{5}{2} = \frac{5}{2}k'$. □

Using Lemma 7.2 and a similar argument as in [19], we can now obtain a bound on the number of edges in RAC_1 graphs in which all faces of G_0 are good. Namely, a planar triangulation has $3n - 6$ edges even in the presence of non-homotopic parallel edges. Hence, G has at most $3n - 6 - k + \frac{8}{3}k$ edges. On the other hand, since $|E_1| \leq 4n - 8$ as shown in [19] and discussed in Section 7.1, we also have that G has at most $7n - 14 - k$ edges. The minimum of both bounds is maximized for $k = \frac{3}{2}(n - 2)$ yielding a bound of $|E| \leq 5.5n - 11$. With an analogous argument, we obtain that $|E| \leq 5.4n - 10.8$ for simple RAC_1 drawings when $k = \frac{8}{5}(n - 2)$.

It remains to prove that it is not a loss of generality to assume that every face of G_0 is good:

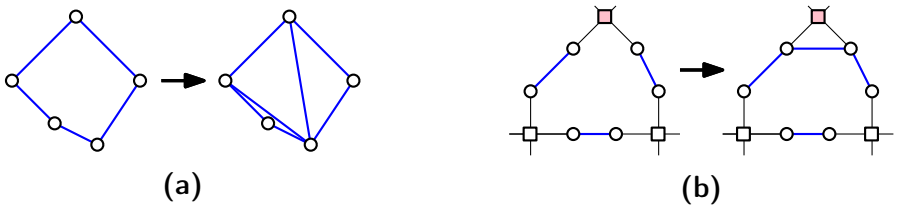


Figure 7.6: Operations of the augmentation procedure described in the proof of Lemma 7.3. Planar edges are drawn blue. Vertices belonging to G are drawn as circles, dummy vertices introduced in the planarization step as boxes. The red dummy vertex in subfigure (b) is the one from which the traversal of the face is started.

Lemma 7.3. *Let G be a RAC$_1$ graph such that not all faces of G_0 are good. Then, G can be augmented to a not necessarily simple supergraph G^* by the introduction of intersection-free edges which are drawn as simple Jordan arcs such that (i) all faces in G_0^* are good, (ii) all parallel edges and self-loops in G_0^* are non-homotopic, and, (iii) in the obtained drawing of G^*, graph G is represented by a RAC$_1$ subdrawing.*

Proof. Consider the planarization G' of G. Let f be a face of G_0 that is not good. Then, there exists a face f' of G' inside f that is not triangular and contains at least two edges from the boundary of f. If f' is bounded only by planar edges, that is, f and f' coincide, we triangulate f'; see Fig. 7.6a. Otherwise, we select an arbitrary dummy vertex on the boundary of f' and traverse the boundary of f'. We then connect the first vertex incident to a planar edge in this traversal with the last vertex incident to a planar edge in this traversal. Note that this can be the same vertex if it is a cut vertex of face f'. The newly introduced edge can always be drawn because we only require it to be represented by a simple Jordan arc. In addition, it splits f' into a face that contains only one planar edge and a second smaller face, hence, this procedure eventually terminates. In the resulting drawing, edges may have any number of bends, however, the subdrawing of G is still the valid RAC$_1$ drawing we started with.

In both cases, the newly introduced edges are not parallel to another edge present in the same face, thus, they are non-homotopic by construction. In the end, every face of the planarization is either a triangle of planar edges or contains at most one planar edge. In other words, every face is good. □

Recall that Lemma 7.2 allows for parallel non-homotopic edges. Hence, we only have to deal with self-loops. As discussed in the proof of Lemma 7.3, such self-loops appear when a vertex v is a cut-vertex of a face of the planarization G'. Note that v then is also a cut-vertex of G.

Consider a self-loop s. Edge s is incident to a cut-vertex v of G and encloses a subdrawing for which we assume that it does not contain any other self-loop. Let

H_i and H_o denote the subgraphs of G induced by the vertices of G in the interior and exterior of s, respectively. In particular, v is part of both H_i and H_o which is not true for any other vertex of G. In addition, s does not belong to H_i or H_o. Let n_i and n_o denote the number of vertices of H_i and H_o, respectively, and let m_i and m_o denote the corresponding number of edges. Since v is accounted for in n_i and n_o, we have that $n = n_i + n_o - 1$. Using induction, we may assume that $m_i \leq 5.5n_i - 11$ and $m_o \leq 5.5n_o - 11$. Then, G contains at most $5.5(n_i + n_o) - 22 + 1 = 5.5n - 15.5 < 5.5n - 11$ edges. Analogously, in the case where the drawing is simple, we have that $m_i \leq 5.4n_i - 10.8$ and $m_o \leq 5.4n_o - 10.8$ and conclude that G has at most $5.4(n_i + n_o) - 21.6 + 1 = 5.4n - 15.2 < 5.4n - 10.8$ edges. Therefore, the upper bound also holds if G_0 contains non-homotopic self-loops.

We conclude with the main theorem of this section:

Theorem 7.1. *Let G be a graph with n vertices that admits a RAC$_1$ drawing. Then, G has at most $5.5n - 11$ edges.*

Similar, for graphs that admit simple RAC$_1$ drawings, we obtain:

Theorem 7.2. *Let G be a graph with n vertices that admits a simple RAC$_1$ drawing. Then, G has at most $5.4n - 10.8$ edges.*

7.3 Lower Bound Results

In this section, we present new lower bound results on the edge density of RAC$_1$ graphs both in the general setting and in the restricted scenario where the drawings have to be simple. We first prove that the result from Theorem 7.1 is tight up to a constant additive factor.

Theorem 7.3. *For every $k \in \mathbb{N}$, there exists a graph with $n = 16k + 32$ vertices and $5.5n - 72$ edges that admits a RAC$_1$ drawing.*

Proof. We define a family of graphs that fulfils the properties stated in the theorem. The most important ingredient of our construction is a *hexagonal tile*, that is, a RAC$_1$ drawing of $K_6 - e$ (the graph obtained from K_6 by removing an edge) which has the property that all vertices are located on the outer face; see Fig. 7.7a. Let $v_1, \ldots, v_6$ denote the vertices of a tile and let $e = (v_2, v_5)$. Then, we can describe the drawing of the tile more precisely as follows:

(i) v_1 is located at $(1, 40)$.

(ii) v_2 is located at $(0, 9)$.

(iii) v_3 is located at $(9, 0)$.

(iv) v_4 is located at $(\frac{1}{97}(1313 + 1770\sqrt{2}), \frac{1}{97}(1960 - 528\sqrt{2}))$.

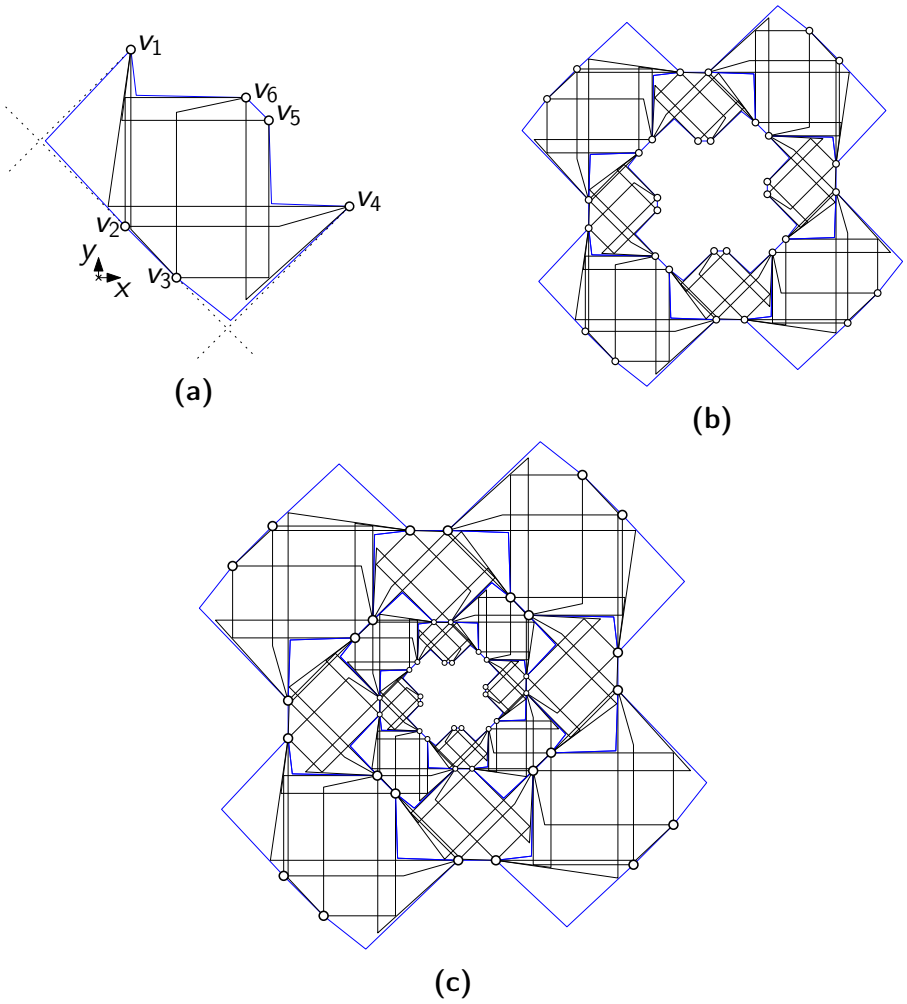


Figure 7.7: Construction of a graph family with $n = 16k+32$ vertices and $5.5n-72$ edges that admits RAC$_1$ drawings. (a) Hexagonal tile. (b) A ring of tiles. (c) Tiling of the plane with nested rings. Planar edges are drawn blue.

(v) v_5 is located at $(\frac{1}{97}(1313 + 800\sqrt{2}), \frac{1}{97}(5880 + 1520\sqrt{2}))$.

(vi) v_6 is located at $(\frac{1}{97}(925 + 800\sqrt{2}), \frac{1}{97}(7044 + 1520\sqrt{2}))$.

(vii) The outer cycle $(v_1, \ldots, v_6)$ is drawn with planar edges.

(viii) Edges (v_2, v_3) and (v_5, v_6) are straight-line. Moreover, the representation of (v_5, v_6) can be obtained from the representation of (v_2, v_3) by a scaling factor of $4/9$.

(ix) Edge (v_1, v_2) has a bend that forms a convex (geometric) vertex of the outer face. This bend is arbitrarily close to the intersection of the line of slope 1 through v_1 and the line of slope -1 through v_2. Moreover, from its representation, it is possible to obtain the representation of (v_4, v_5) by a scaling

factor of $2/3$ and a rotation by $\pi/4$. Consequently, the bend of (v_4, v_5) forms a reflex (geometric) vertex of the outer face.

(x) Edge (v_3, v_4) has a bend that forms a convex (geometric) vertex of the outer face. This bend is arbitrarily close to the intersection of the line of slope -1 through v_3 and the line of slope 1 through v_4. Moreover, from its representation, it is possible to obtain the representation of (v_1, v_6) by a scaling factor of $2/3$ and a rotation by $-\pi/4$. Consequently, the bend of (v_1, v_6) forms a reflex (geometric) vertex of the outer face.

(xi) All remaining edges have a bend and are involved in intersections that occur on horizontal and vertical segments.

Due to the symmetry of edges (v_1, v_2) and (v_4, v_5) as well as edges (v_3, v_4) and (v_1, v_6), it is possible to create a *ring of tiles* as shown in Fig. 7.7b. More precisely, such a ring is composed of four tiles of unit size rotated by 0, $\pi/2$, π and $3\pi/2$, respectively, and four tiles scaled by a factor of $2/3$ rotated by $\pi/4$, $3\pi/4$, $5\pi/4$ and $7\pi/4$, respectively. A ring of tiles has 32 vertices; 16 of them are located on the outer face while the remaining 16 are incident to the innermost face. Also, the ring of tiles clearly admits a RAC_1 drawing.

Due to the properties of the edges bounding the tiles, we also observe that the polygon bounding the innermost face can be obtained from the polygon bounding the outer face by scaling it by a factor of $4/9$. Consequently, we can take two copies of a ring, one of unit size and one scaled by a factor of $4/9$, and identify the innermost face of the first with the outer face of the second copy; see Fig. 7.7c. The obtained graph has 16 more vertices than a ring of tiles and still admits a RAC_1 drawing. Moreover, the innermost face still has the same geometry up to scaling so that the outer face of a new ring can be identified with it. This gives rise to a family of graphs with $16k + 32$ vertices for $k \in \mathbb{N}$ that admit RAC_1 drawings.

Regarding the density of this graph family, we observe that in the subgraph induced by the planar edges all faces except the innermost and outer face are hexagons. The two remaining faces have length 16. By Euler's formula the subgraph induced by the planar edges then has $\frac{3}{2}n-8$ edges and $\frac{1}{2}n-6$ faces. Since in each of the $\frac{1}{2}n-8$ hexagonal faces eight additional intersected edges are present, we conclude that graphs of this family have $\frac{3}{2}n - 8 + 8(\frac{1}{2}n - 8) = 5.5n - 72$ edges. □

We point out that our lower bound construction in the proof of Theorem 7.3 is closely related to the 3-planar graphs of maximum edge density [38]. In fact, the subgraph induced by the planar edges of a 3-planar graph of maximum edge density has only hexagonal faces (however, parallel edges may be present) and in each hexagonal face eight chords are present. In fact, the construction even omits the same chord in each face as we did in the construction of the tiles, however, the drawing of the tile is not 3-planar; for instance edge (v_1, v_5) in Fig. 7.7a is

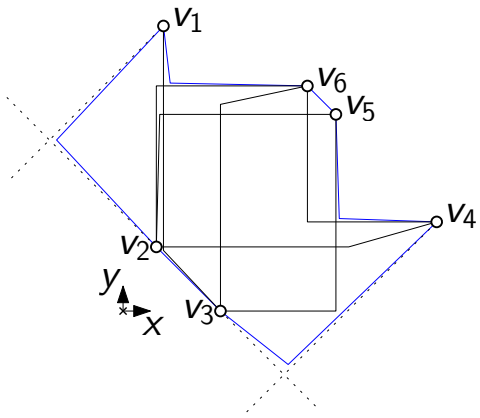


Figure 7.8: A simple RAC$_1$ drawing of $K_6 - \{e, e'\}$ with the same outer face as the tile in Fig. 7.7a.

intersected four times. According to their edge density, the graph family described in the proof of Theorem 7.3 does not admit simple RAC$_1$ drawings. In fact, in each tile, edges (v_4, v_6) and (v_4, v_2) intersect. We could easily adopt our proof for the case where simple drawings are required, as $K_6 - \{e, e'\}$, the graph obtained from K_6 by removing two edges, admits a simple RAC$_1$ drawing where the outer face has the same shape as the tile in Fig. 7.7a. This would suffice to prove a lower bound of $5n - 64$ for the edge density of simple RAC$_1$ graphs. However, we will show a slightly stronger result in the next theorem. Namely, we show that some of the 2-planar [38], fan-planar [118] and gap-planar [22] graphs of maximum edge density also admit simple RAC$_1$ drawings.

Theorem 7.4. *For every $k \in \mathbb{N}$, there exists a graph with $n = 15k + 20$ vertices and $5n - 10$ edges that admits a simple RAC$_1$ drawing.*

Proof. Similar to the lower bound constructions for the maximum edge density of 2-planar [38], fan-planar [118] and gap-planar [22] graphs, we describe a graph family that consists of a planar spanning subgraph whose faces are pentagons and all chords inside each face which intersect as shown in Fig. 7.9c. The basic component of our construction is the dodecahedral graph which admits a planar straight-line drawing where all faces have one out of three different geometric shapes (up to scaling); see Fig. 7.9a. In particular, the outer face and the innermost face (colored red in Fig. 7.9a) are regular pentagons while all five faces incident to the outer face have the same geometry (up to rotation). In addition, the five faces incident to the innermost face also have the same geometry (but differrent than the geometry of the faces incident to the outer face). Moreover, each face is symmetric to a line that is perpendicular to one of its sides (length denoted by a in Fig. 7.9b) and passes through the opposite vertex of the face (denoted by A in Fig. 7.9b). Using the notation of Fig. 7.9b, we can more precisely define the geometry of the drawing:

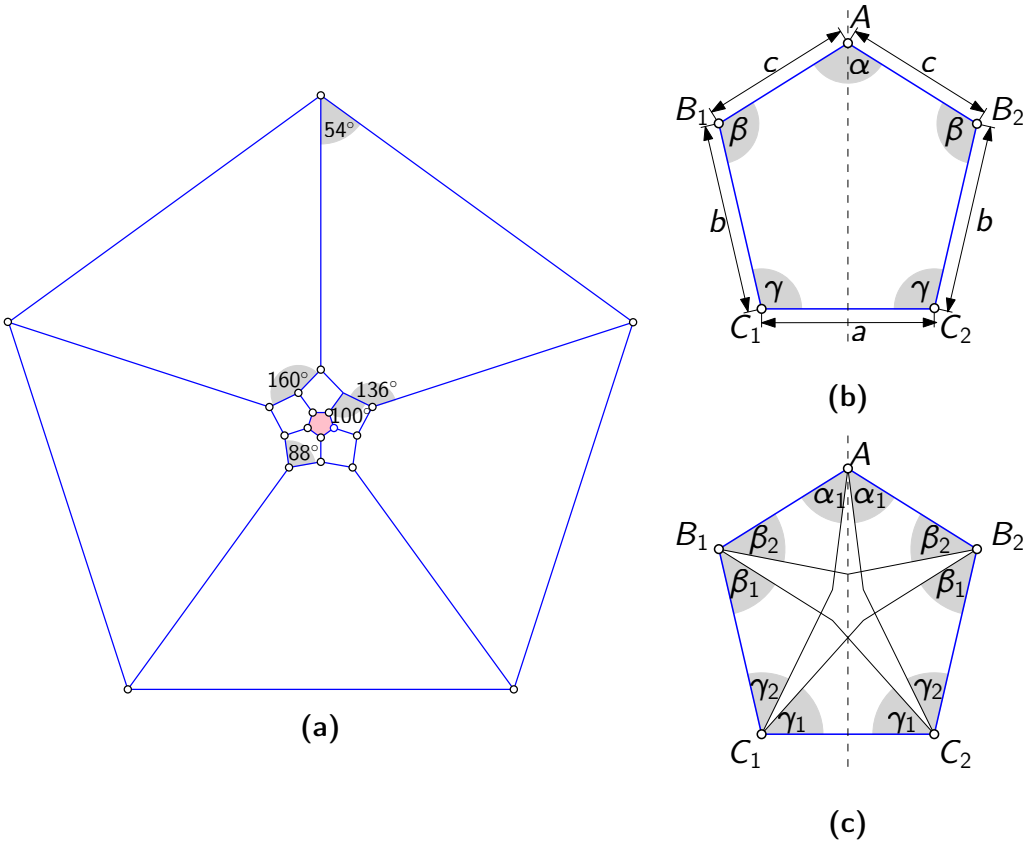


Figure 7.9: Overview of our lower bound construction for simple RAC_1 graphs. (a) Drawing of the dodecahedral graph. (b) Notation of edge lengths and angles in a pentagonal face. (c) Intersection configuration and notations for angles formed by intersecting edges. Planar edges are colored blue.

(i) For the faces incident to the innermost face, the side of length a is shared with the innermost face. Moreover, $\alpha = 88°$, $\beta = 100°$ and $b = 1.5a$.

(ii) For the faces incident to the outer face, the side of length a is shared with the outer face. Moreover, $\alpha = 160°$, $\gamma = 54°$ and $b = 8.5c$.

Since both the outer and the innermost face are regular pentagons, it is possible to merge two copies of the dodecahedral graph by identifying the outer face of one copy with the innermost face of the second copy. We can repeat the same operation with the newly obtained graph as the innermost face is still a regular pentagon. Repeating this process k times yields a graph with $n = 15k+20$ vertices since the dodecahedral graph has 20 vertices and 5 vertices are shared between consecutive copies. The resulting family of graphs admits drawings in which the faces of the planar subgraph only have the three geometries discussed above (up to scaling and rotation).

It remains to discuss the five chords inside each pentagonal face. In each

face the following property is maintained: A chord intersects exactly two other chords and bends in between both intersections; see Fig. 7.9c. As a result, since we already defined the face geometries above, it suffices to provide the angles as labeled in Fig. 7.9c to uniquely define the drawing:

(i) In the innermost face, $\alpha_1 = \beta_1 = \beta_2 = \gamma_1 = \gamma_2 = 45°$; see Fig. 7.10a.

(ii) In the outer face, $\alpha_1 = \beta_1 = \beta_2 = \gamma_1 = \gamma_2 = 45°$; see Fig. 7.10b.

(iii) In a face incident to the innermost face, $\alpha_1 = 40°$, $\beta_1 = 30°$, $\beta_2 = 50°$, $\gamma_1 = 45°$ and $\gamma_2 = 60°$; see Fig. 7.10c.

(iv) In a face incident to the outer face, $\alpha_1 = 47.5°$, $\beta_1 = 85°$, $\beta_2 = 42.5°$, $\gamma_1 = 45°$ and $\gamma_2 = 5°$; see Fig. 7.10d.

We conclude that each graph in the family admits a simple RAC$_1$ drawing. By Euler's formula the subgraph induced by the planar edges has $\frac{5}{3}(n-2)$ edges and $\frac{2}{3}(n-2)$ faces. Since in each face five additional intersected chords are present, we conclude that graphs of this family have $\frac{5}{3}(n-2) + 5 \cdot \frac{2}{3}(n-2) = 5n - 10$ edges. □

We note that both of our lower bound constructions use exponential area.

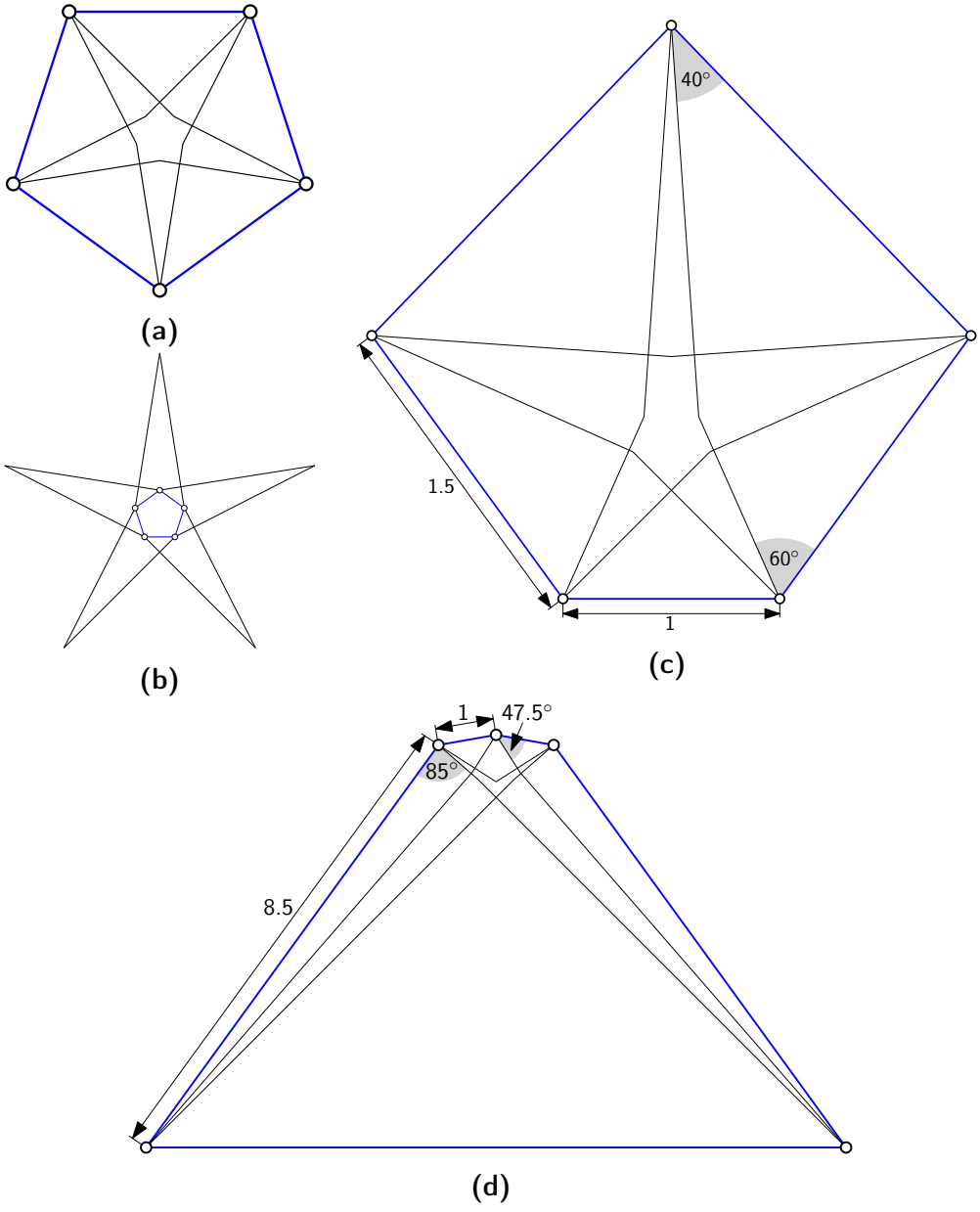


Figure 7.10: Drawings of chords inside (a) the innermost face, (b) the outer face, (c) faces incident to the innermost face, and, (d) faces incident to the outer face. Planar edges are drawn blue.

Chapter 8

Area Bounds for RAC Drawings with three and more Bends per Edge

Many graphs that appear in real-world applications are non-planar. Experimental evaluation of the human perception of graph drawings has shown that non-planar drawings with a large crossing resolution are better readable than those where the crossing resolution is small [113, 115]. This observation gave rise to the study of RAC drawings, in which all intersections occur at right angles [71]. The curve complexity of RAC drawings is typically also taken into account, since it is known to negatively correlate with readability [140, 141]. The initial paper on RAC drawings [71] considered RAC_3 drawings along with RAC drawings of lower curve complexity and proved that every graph admits a RAC_3 drawing. It is easy to verify that the corresponding drawing algorithm produces drawings in $\Theta(n^4)$ area for every simple graph on n vertices even though this is not explicitly stated in [71].

In several follow-up studies, the area of RAC drawings has become a main focus. In this line of research, it was shown that there are planar graphs that still require quadratic area in any RAC_0 drawing [14] while NIC-planar and general 1-planar graphs admit polynomial area RAC_1 and RAC_2 drawings, respectively [55]. Regarding more general graph classes, it was shown that every simple graph admits a RAC_4 drawing in $\Theta(n^3)$ area [66]. In the same paper, it was proven that every simple graph admits a drawing in $\Theta(n^2(\cot \varepsilon/2)^2)$ area with one bend per edge under the relaxation that every intersection occurs at an angle of at least $\pi/2 - \varepsilon$ for $\varepsilon > 0$. Such drawings are known as large angle crossing drawings and for fixed ε, this result is asymptotically optimal. Note that the multiplicative term $\cot \varepsilon/2$ can however be quite restrictive for moderate values of n while no true right angles can be achieved with the construction. With respect to RAC drawings of complete graphs, the best known area bound was presented in [142] where it was shown

that each simple graph admits a RAC$_6$ drawing in $\Theta(n^{2.75})$ area.

In the analysis of the area, vertices and bends of edges are assumed to be positioned on an integer grid whereas intersections may also occur on non-grid points. It is noteworthy that the positions of intersections usually do not need to be calculated explicitly as they are implied by the positions of the endpoints of intersected edges. The area of RAC drawings of general graphs is trivially in $\Omega(n^2)$ since only graphs of limited edge density can be drawn without bends [71], that is, in graphs with $\Theta(n^2)$ edges, there are $\Theta(n^2)$ bends that must be assigned to distinct grid points. We emphasize that a RAC drawing in $\Theta(n^2)$ area cannot be trivially computed by adding more bends since each such bend must be assigned a distinct integer coordinate. In addition, we assume in the following that graphs are simple since every parallel edge would contribute another bend. Hence, in graphs where every edge is parallel to at most k edges, the area lower bound is $\Omega(kn^2)$.

In this chapter[1], we present several new results on the area requirements of RAC drawings of dense graphs. In Section 8.1, we prove that every graph admits a RAC$_3$ drawing in $\mathcal{O}(n^3)$ area. In contrast, we show in Section 8.2 that not every graph admits a RAC$_3$ drawing in $\mathcal{O}(n^2)$ area. Then, in Section 8.3, we prove that a RAC$_8$ drawing in quadratic area exists for every graph. Finally, we investigate the special case where the graph is p-partite in Section 8.4 and show that even a RAC$_3$ drawing in $\mathcal{O}(n^2p^4)$ area can be achieved.

Before we prove our result, we first establish some notation used throughout this chapter. In all proofs we assume w.l.o.g. that every edge has the same number of bends (which may however occur at angle π). This allows us to distinguish several unique types of segments along an edge. Namely, every edge (u, v) has two *start segments* which are the two segments incident to the endpoints u and v. In the case of RAC$_3$ drawings, we call the remaining segments *middle segments* and the bend connecting both middle segments the *middle bend*. We also call a start segment that is incident to a vertex v a *start segment of* v. In our upper bound constructions, start segments will be located in regions that are free of intersections, which we refer to as *start regions*.

8.1 A New Upper Bound for the Area of RAC$_3$ Drawings

In this section, we show that every graph admits a RAC$_3$ drawing in $\mathcal{O}(n^3)$ area. This result improves upon two prior results: Namely, it was known that every graph admits a RAC$_3$ and a RAC$_4$ drawing in $\mathcal{O}(n^4)$ and $\mathcal{O}(n^3)$ area, respectively [66, 71].

Theorem 8.1. *Let G be a simple graph with n vertices. Then, G admits a RAC$_3$ drawing in $\mathcal{O}(n^3)$ area.*

[1]The results of this chapter also appeared in [94].

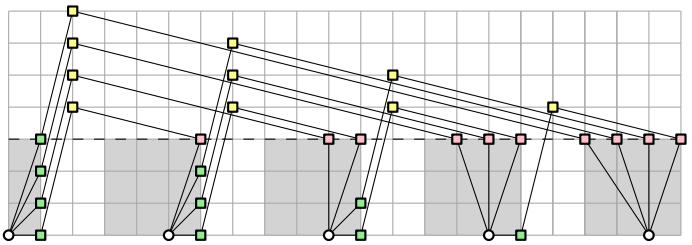


Figure 8.1: A RAC$_3$ drawing of K_5 produced by the algorithm in the proof of Theorem 8.1 in 21×7 area. Vertices are drawn as circles, bends of edges as squares.

Proof. We describe an algorithm that computes a RAC$_3$ drawing in $\mathcal{O}(n^3)$ area. In particular, we describe how to draw the complete graph on n vertices; see Fig. 8.1 for an illustration of the drawing of K_5. Similar to previously known constructions [66, 71], each vertex and its incident segment are located in a distinct region of quadratic area; see the gray regions in Fig. 8.1. Similar to the proof in [66], we distinguish two types of bends incident to a vertex, namely those bends that connect to a vertex with smaller index and those that connect to a vertex with larger index; see the red and green colored squares in Fig. 8.1, respectively. The middle segments are of almost horizontal and vertical slope.

More precisely, consider an arbitrary enumeration $(v_0, \ldots, v_{n-1})$ of the vertices of K_n. Vertex v_i for $0 \leq k \leq n-1$ is placed at $(in, 0)$; see the white circles in Fig. 8.1. Consider an edge (v_i, v_j) with $i < j$. Let $a_{i,j}$ denote the bend of (v_i, v_j) incident to v_i, $b_{i,j}$ denote the middle bend of (v_i, v_j) and $c_{i,j}$ denote the bend of (v_i, v_j) incident to v_j. We position

(i) $a_{i,j}$ at $(in+1, j-i-1)$; see the green squares in Fig. 8.1.

(ii) $b_{i,j}$ at $(in+2, n+j-i-2)$; see the yellow squares in Fig. 8.1.

(iii) $c_{i,j}$ at $(jn-j+i+2, n-2)$; see the red squares in Fig. 8.1.

First consider the start segments. Consider a vertex v_i. We observe that all start segments between v_i and bends $a_{i,j}$ for $j > i$ and bends $c_{k,i}$ for $k < i$ are located in a rectangular start region, ranging from $(i-1)n+3$ to $in+1$ in x-direction and from 0 to $n-2$ in y-direction; see the gray regions in Fig. 8.1. Since all bends $a_{i,j}$ and $c_{k,i}$ are disjoint and located on the boundary of the start region, the start segments incident to v_i are intersection-free. Moreover, the start regions of vertices v_i and v_j for $j > i$ do not overlap since $(j-1)n+3 > in+1$ even if $j = i+1$.

It is easy to verify that the middle segment between bends $a_{i,j}$ and $b_{i,j}$ of edge (v_i, v_j) has slope $n-1$. Moreover, such a segment does not intersect the start region of vertex v_k for $k > i$, since $(k-1)n+3$ (the leftmost x-coordinate of the start region of vertex v_k) is strictly larger than $in+2$ (the x-coordinate of $b_{i,j}$) even if $k = i+1$.

In addition, by construction, the middle segment between bends $b_{i,j}$ and $c_{i,j}$ of edge (v_i, v_j) has slope $-1/(n-1)$. Note that these middle segments are perpendicular to the previous type of middle segments. Since the y-coordinate of $c_{i,j}$ is $n-2$, such segments do not intersect start regions. Hence, only middle segments may intersect. Since the intersections of middle segments with the horizontal line at $y = n-2$ are distinct (see the dashed line in Fig. 8.1), no two parallel segments overlap. Thus the drawing is indeed a RAC drawing.

The smallest x- and y-coordinates are both 0 while the corresponding largest coordinates are n^2-n+1 and $2n-3$, respectively, hence the area bound follows. □

8.2 A First Lower Bound for the Area of RAC$_3$ Drawings

In this section, we show that not all graphs admit a RAC$_3$ drawing in $\mathcal{O}(n^2)$ area. Our proof is by contradiction and can be outlined as follows: First, in Lemma 8.1 we show that in any RAC drawing with few bends per edge, there exist two sets of edge segments S_i and T_i of cardinality $\Omega(n^2)$ such that there are $\Omega(n^4)$ intersections between segments in S_i and T_i. Then, we derive geometric properties of such segments in Lemmas 8.2 and 8.3. Afterwards, we consider properties that are specifically true for RAC$_3$ drawings and show in Lemmas 8.4 to 8.6 that there are $\Omega(n^2)$ edges that have both a segment from S_i and a segment from T_i. We continue by subdividing the drawing into a set of disjoint regions $\mathcal{R}_i$ which contain only at most one endpoint of each segment in S_i and T_i. In Lemmas 8.7 and 8.8, we restrict the possible positions of vertices connected to endpoints of segments from S_i and T_i located in a region of $\mathcal{R}_i$. This allows us to show in Lemma 8.9, that the edges which contain both a segment from S_i and from T_i induces a subgraph which is nearly p-partite for some integer p except for a linear number of edges which we call *odd edges*. As a result, in the proof of Theorem 8.2, we show that any RAC$_3$ drawing of K_n in quadratic area contains a drawing of a complete subgraph that has too few edges which have a segment from S_i and a segment of T_i; a contradiction to Lemma 8.6 for the drawing of the subgraph.

Lemma 8.1. *Let Γ be a RAC$_b$ drawing of a graph G with n vertices and $\Omega(n^2)$ edges for $b = \mathcal{O}(1)$. Then, in Γ, there are two sets of parallel edge segments S_i and T_i with $|S_i|, |T_i| = \Omega(n^2)$ such that segments in S_i are perpendicular to segments in T_i.*

Proof. Recall that in a RAC drawing all intersections appear between two perpendicular edge segments and that, by the crossing lemma [7, 124], any drawing of G contains $\Omega(n^4)$ intersections. We partition the segments in Γ that are involved in intersections based on their slope into maximal sets $S_0, \ldots, S_k$ and $T_0, \ldots, T_k$ such that the segments in S_i and T_i are perpendicular for $0 \leq i \leq k$. Assume

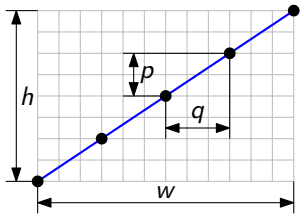


Figure 8.2: A fine-horizontal grid line (blue) and its common points with the coarse grid (gray).

w.l.o.g. that $|S_i| \geq |T_i|$ and $|T_i| \geq |T_{i+1}|$. Because every edge has k bends, the number of segments assigned to one of the sets S_i is upper bounded by sn^2 for some constant $s = f(k)$, that is,

$$|S_0| + \sum_{i=1}^{k} |S_i| \leq sn^2 \quad \text{or} \quad |S_0| \leq sn^2 - \sum_{i=1}^{k} |S_i|.$$

Since intersections occur only between pairs of segments belonging to S_i and T_i for some i, we can bound the number of intersections $cr(\Gamma)$ in Γ as follows:

$$\begin{aligned} cr(\Gamma) \leq |S_0||T_0| + \sum_{i=1}^{k} |S_i||T_i| &\leq \left(sn^2 - \sum_{i=1}^{k} |S_i|\right)|T_0| + \sum_{i=1}^{k} |S_i||T_i| \\ &= sn^2 - \sum_{i=1}^{k} (|T_0| - |T_i|)|S_i| \leq sn^2|T_0|. \end{aligned}$$

The term $sn^2|T_0|$ has to be in $\Omega(n^4)$ which is the case if $|T_0| = \Omega(n^2)$. This then also implies $|S_0| = \Omega(n^2)$. □

In the following, we consider all maximal sets of parallel edge segments which are involved in $\Omega(n^4)$ intersections. We partition these sets into families $\mathcal{S} = \{S_1, \ldots, S_k\}$ and $\mathcal{T} = \{T_1, \ldots, T_k\}$ such that the segments in S_i are perpendicular exactly to the segments in T_i.[2] Observe that in a RAC$_b$ drawing for constant b, the maximum index k is a constant.

Consider a pair of segment sets $S_i \in \mathcal{S}$ and $T_i \in \mathcal{T}$. In the following analysis and all illustrations in this section, we assume w.l.o.g. that the slope of segments in S_i is positive. First, we will derive properties of the slope s_i of segments belonging to segment set $S_i \in \mathcal{S}$. In particular, we use the fact that segments in S_i and T_i follow the grid lines of a *fine* grid $\mathcal{F}_i$ that is finer than the *coarse* integer grid on which vertices and bends are located; see Fig. 8.2. More precisely, the endpoints of each segment is an integer coordinate on the coarse grid. This directly implies that s_i is a rational number and that the intersections between segments in S_i

[2] In contrast to the proof of Lemma 8.1, we now only consider sets of edge segments that are involved in many intersections.

and T_i are located on rational coordinates located on a fine grid $\mathcal{F}_i$. Formally, the fine grid $\mathcal{F}_i$ is defined by *fine-horizontal* grid lines of slope s_i and *fine-vertical* grid lines having slope of slope $-1/s_i$ that pass through at least two points of the coarse grid; see Fig. 8.2. Let $s_i = p/q$ for coprime integer p and q. By scaling the drawing by factor pq, the intersections between segments in S_i and T_i are located on integer coordinates. Since there are $\Omega(n^4)$ intersections in the drawing, this scaling factor cannot be arbitrarily small for small drawing areas. We analyze this property in the next lemma:

Lemma 8.2. *Let Γ be a RAC_b drawing of a graph G with n vertices and $\Omega(n^2)$ edges for $b = \mathcal{O}(1)$ with width w and height h. In addition, let $s_i = p/q$ be the slope of segments in $S_i \in \mathcal{S}$ for coprime integers p and q. Then*

1. $\max\{p, q\} \in \Omega(\sqrt{n^4/wh})$ *or* $pq \in \Omega(n^4/\max\{w^2, h^2\})$, *and,*

2. $p, q \in \mathcal{O}(\min\{w, h\})$.

Proof. Depending on the values of p and q, we observe, that fine grid lines might pass through more than two points of the coarse grid; see Fig. 8.2. This however limits how many fine grid lines exist.

Consider two consecutive fine-horizontal grid lines ℓ_1 and ℓ_2. For $i \in \{1, 2\}$, the line ℓ_i can be expressed by a formula of form $y = p/q \cdot x + b_i$. Since all lines pass through an integer point, it follows that $b_i = c_i/q$ for some integer c_i. Since ℓ_1 and ℓ_2 are consecutive, $|c_2 - c_1| = 1$. This means, that the vertical distance between ℓ_1 and ℓ_2 is equal to $1/q$. Next, consider their horizontal distance. For this purpose, we set $p/q \cdot x_1 + b_1 = p/q \cdot x_2 + b_2$ which yields $|x_2 - x_1| = q/p|b_2 - b_1| = 1/p|c_2 - c_1| = 1/p$, that is, their horizontal distance is $1/p$. Analogously, the horizontal and vertical distance between two consecutive fine-vertical grid lines is $1/q$ and $1/p$, respectively.

We conclude that there are at most $\Theta(\max\{wp, hq\})$ fine-horizontal and $\Theta(\max\{wq, hp\})$ fine-vertical grid lines. The number of intersections between these grid lines is in $\Theta(\max\{w^2pq, whp^2, whq^2, h^2pq\})$ which must be $\Omega(n^4)$ by the crossing lemma [7, 124]. This is the case if $\max\{p, q\} \in \Omega(\sqrt{n^4/wh})$ or $pq \in \Omega(n^4/\max\{w^2, h^2\})$, yielding Claim 1.

Finally, since the endpoints of each segment are on the coarse grid, it follows that $h, w \geq \max\{p, q\}$ which implies Claim 2. □

All previous lemmas hold without any restriction on the area of the corresponding RAC drawing. We now refine Lemma 8.2 for the case where the drawing area is quadratic.

Lemma 8.3. *Let Γ be a RAC_b drawing of a graph G with n vertices and $\Omega(n^2)$ edges for $b = \mathcal{O}(1)$ with width w and height h in $\mathcal{O}(n^2)$ area. In addition, let $s_i = p/q$ be the slope of segments in $S_i \in \mathcal{S}$ for coprime integers p and q. Then*

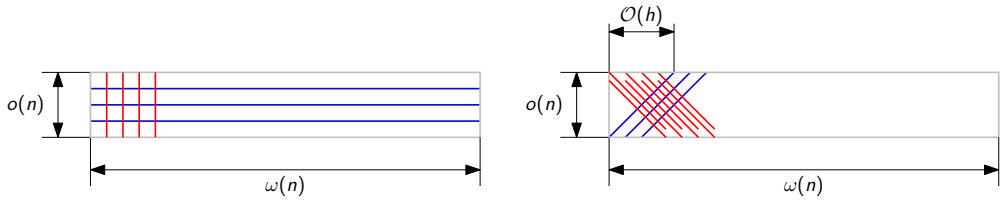


Figure 8.3: If the drawing area is $\omega(n) \times o(n)$, either (a) the fine grid $\mathcal{F}_i$ is equal to the coarse grid, or, (b) fine-horizontal grid lines intersect $\mathcal{O}(h^2) = o(n^2)$ fine-vertical grid lines (red) each.

1. $h, w \in \Theta(n)$, *and,*

2. $\max\{p, q\} \in \Theta(n)$.

Proof. Assume for a contradiction that $h = o(n)$ which implies that $w = \omega(n)$. By Lemma 8.2,

1. $\max\{p, q\} \in \Omega(n)$ or $pq \in \Omega(h^2)$, and,

2. $p, q \in \mathcal{O}(h)$.

Due to Property 2, we observe that Property 1 can only be fulfilled by condition $pq = \Omega(h^2)$. Consider the fine grid $\mathcal{F}_i$. If the fine-horizontal grid lines are indeed horizontal, that is, $p = 0$, the fine and the coarse grid are identical; see Fig. 8.3a. However, there are only $\mathcal{O}(n^2)$ intersections in the coarse grid which is less than $\Omega(n^4)$, the number of intersections by the crossing lemma [7, 124].

Otherwise, $p \neq 0$. Because $pq \in \Omega(h^2)$ and $p, q \in \mathcal{O}(h)$, it follows that $p, q \in \Theta(h)$. Then, fine-horizonal grid lines have only length $\mathcal{O}(h)$ in the drawing area and can only be intersected by $\Omega(h^2)$ fine-vertical grid lines each; see Fig. 8.3b. Since $h = o(n)$, this yields $o(n^4)$ intersections in total as there are only $\Theta(n^2)$ fine-horizontal grid lines; a contradiction.

We conclude that our initial assumption $h = o(n)$ was wrong and that therefore $h, w = \Theta(n)$ which directly yields $p, q \in \Theta(n)$. □

Next, we consider how the segments of $\mathcal{S}$ and $\mathcal{T}$ can be distributed on the edges of G. For this purpose, consider sets $S_i \in \mathcal{S}$ and $X_j \in \mathcal{S} \cup \mathcal{T}$ where $X_j = S_j$ if it belongs to $\mathcal{S}$; otherwise $X_j = T_j$. We denote by $E[S_i, \bar{X}_j]$ the set of edges with a segment from S_i but no segment from $X_j \neq S_i$. Moreover, we denote by $E[S_i, X_j]$ the set of edges with both a segment from S_i and a segment from $X_j \neq S_i$. Similarly, we can define $E[\bar{X}_j, T_i]$ and $E[X_j, T_i]$, which only amounts to renaming $\mathcal{S}$ and $\mathcal{T}$. For an illustration refer to Fig. 8.4 where we color segments from S_i blue and segments from T_i red; a color scheme we will use throughout this section. We will show that there are sets S_i and T_i such that $|E[S_i, T_i]| = \Omega(n^2)$

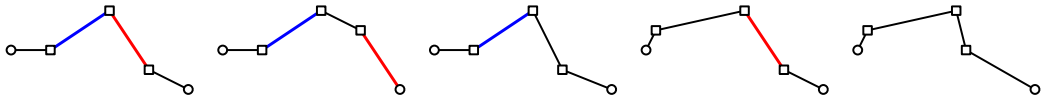


Figure 8.4: (a)–(b) Edges belonging to $E[S_i, T_i]$, (c) edge belonging to $E[S_i, \bar{T}_i]$, (d) edge belonging to $E[\bar{S}_i, T_i]$, and, (e) edge belonging to none of $E[S_i, \bar{T}_i]$, $E[\bar{S}_i, T_i]$ and $E[S_i, T_i]$. Segments belonging to S_i are drawn blue, segments belonging to T_i are drawn red.

if the drawing is a RAC$_3$ drawing. First, we show an intermediate statement for RAC$_b$ drawings with constant b:

Lemma 8.4. *Let Γ be a RAC$_b$ drawing of a graph G with n vertices and $\Omega(n^2)$ edges for $b = \mathcal{O}(1)$. Then, in Γ, there is a pair of segment sets $S_i \in \mathcal{S}$ and $X_j \in \mathcal{S} \cup \mathcal{T}$ with $|E[S_i, X_j]| = \Omega(n^2)$ or a pair of segment sets $T_i \in \mathcal{T}$ and $X_j \in \mathcal{S} \cup \mathcal{T}$ with $|E[X_j, T_i]| = \Omega(n^2)$.*

Proof. If $|E[S_i, X_j]| = \Omega(n^2)$ for some $S_i \in \mathcal{S}$ and $X_j \in \mathcal{S} \cup \mathcal{T}$ with $i \neq j$ or if $|E[X_j, T_i]| = \Omega(n^2)$ for some $T_i \in \mathcal{T}$ and $X_j \in \mathcal{S} \cup \mathcal{T}$ with $i \neq j$, the lemma trivially holds. Thus, assume in the following that $|E[S_i, X_j]| = o(n^2)$ for all pairs $S_i \in \mathcal{S}$ and $X_j \in \mathcal{S} \cup \mathcal{T}$ with $i \neq j$ and $|E[X_j, T_i]| = o(n^2)$ for all pairs $T_i \in \mathcal{T}$ and $X_j \in \mathcal{S} \cup \mathcal{T}$ with $i \neq j$. Assume for a contradiction that also $|E[S_i, T_i]| = o(n^2)$ for all $1 \leq i \leq k$. Then, for each i, set $E[S_i, T_i]$ participates in $o(n^4)$ intersections. Assume w.l.o.g. that $|\bigcup_{i=1}^{k} E[S_i, \bar{T}_i]| \geq |\bigcup_{i=1}^{k} E[\bar{S}_i, T_i]|$ and consider the graph $G' = G \setminus \bigcup_{i=1}^{k} E[\bar{S}_i, T_i]$. Since $|E[S_i, X_j]| = o(n^2)$ for all pairs $S_i \in \mathcal{S}$ and $X_j \in \mathcal{S} \cup \mathcal{T}$ with $i \neq j$, G' still has $\Omega(n^2)$ edges. Moreover, there is a valid subdrawing Γ' of G' in Γ. However, this subdrawing has only $o(n^4)$ intersections, a contradiction to the crossing lemma [7, 124]. □

Until now we considered properties of RAC$_b$ drawings for constant b. The remaining results in this section will be specific for the case where $b = 3$. As a first step, we will improve Lemma 8.4. To do so, we observe which segments can actually belong to sets S_i and T_i:

Observation 8.1. *Each vertex can only be incident to two start segments of the same slope. Thus, there are only $\mathcal{O}(n)$ start segments overall with the same slope.*

Based on this observation, we can show the following:

Lemma 8.5. *Let Γ be a RAC$_3$ drawing of a graph G with n vertices and $\Omega(n^2)$ edges. Then, for $i \neq j$, it holds that $|E[S_i, X_j]| = o(n^2)$ and $|E[X_j, T_i]| = o(n^2)$ with $S_i \in \mathcal{S}$, $T_i \in \mathcal{T}$ and $X_j \in \mathcal{S} \cup \mathcal{T}$.*

Proof. Assume that $|E[S_i, T_j]| = \Omega(n^2)$ for $i \neq j$. By Observation 8.1, for $\Omega(n^2)$ of the edges in $E[S_i, T_j]$ the two middle segments belong to S_i and T_j while the start

segments have different slopes. Let $E'[S_i, T_j]$ denote this set of edges. Consider the start segments of $E'[S_i, T_j]$. Let $\mathcal{P}_{start} = \{P_1, \ldots, P_r\}$ be a partitioning of the start segments into maximal sets of parallel segments for some $r = \mathcal{O}(n^2)$. By Observation 8.5, each set $P_\ell \in \mathcal{P}_{start}$ has cardinality $\mathcal{O}(n)$ and hence there are only $\mathcal{O}(n^2)$ intersections between start segments. In addition, if there are segments in $\mathcal{P}_{start}$ that are perpendicular to S_i or T_j, those are only $\mathcal{O}(n)$ segments which form at most $\mathcal{O}(n^2)$ intersections with middle segments from $E'[S_i, T_j]$.

Now consider the subgraph G' induced by $E'[S_i, T_j]$. G' has $\Omega(n^2)$ edges. Also, drawing Γ contains a valid subdrawing Γ' of G'. However, since segments belonging to S_i and T_j are not perpendicular and may not intersect each other, its drawing Γ' contains only the $\mathcal{O}(n^3)$ intersections which involve start segments; a contradiction to the crossing lemma [7, 124]. □

We summarize Lemmas 8.4 and 8.5 as follows:

Lemma 8.6. *Let Γ be a RAC$_3$ drawing of a graph G with n vertices and $\Omega(n^2)$ edges. Then, in Γ, there is a pair of segment sets $S_i \in \mathcal{S}$ and $T_i \in \mathcal{T}$ with $|E[S_i, T_i]| = \Omega(n^2)$. More precisely, for sufficiently large n, $|E[S_i, T_i]| \geq c_{ST} n^2$.*

We now investigate which connections can be actually realized with edges belonging to $E[S_i, T_i]$ for a pair of perpendicular segment sets $S_i \in \mathcal{S}$ and $T_i \in \mathcal{T}$. Let p_i/q_i be the slope of segments in S_i for coprime integers p_i and q_i. Further, let $\mathcal{R}_i$ be a checkerboard partitioning of the drawing area into axis-parallel square-shaped disjoint regions of side length $\max\{p_i, q_i\}/2$.[3] Note that there are $\mathcal{O}(1)$ regions in $\mathcal{R}_i$ because $\max\{p_i, q_i\} = \Theta(n)$ and $h, w = \Theta(n)$ due to Lemma 8.3. Due to the choice of the slope of segments in S, two consecutive integer points hit by one segment from S are multiples of q_i in horizontal and multiples of p_i in vertical direction apart from each other. Therefore, we observe the following:

Observation 8.2. *Let $s \in S_i \cup T_i$. Then at most one endpoint of s is located in any region $R \in \mathcal{R}_i$. Moreover, if an endpoint of s is located in $R \in \mathcal{R}_i$, s intersects the boundary of R.*

Recall Observation 8.1. Since there are only $\mathcal{O}(n)$ start segments that belong to S_i and T_i, there are $\mathcal{O}(n^2)$ edges that have both a middle segment from S_i and a middle segment from T_i; see Fig. 8.5a. In particular, we will consider the bends that are not middle bends. We refer to bends that are incident to a start segment and a middle segment from S_i as an *S_i-endpoint*; similarly, we call a bend incident to a middle segment from T_i and a start segment as a *T_i-endpoint*; see the marked bends in Fig. 8.5a. Since the segments from S_i are perpendicular to the segments from T_i, we observe the following:

[3] Observe that the regions at the boundary of the drawing may slightly extend beyond the area.

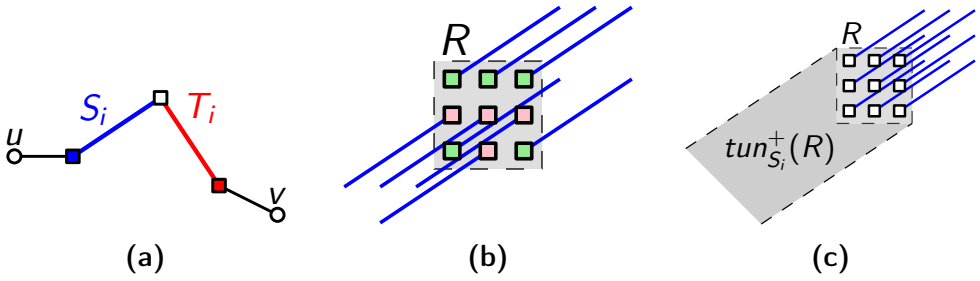


Figure 8.5: (a) An edge (u, v) with middle segments belonging to a perpendicular pair of edge segments $S_i \in \mathcal{S}$ (blue) and $T_i \in \mathcal{T}$ (red). The S_i-endpoint of (u, v) is colored blue; the T_i-endpoint is colored red. (b) A region R and the contained bends belonging to $ep^+_{S_i}(R)$ (green) and $ep^-_{S_i}(R)$ (red). (c) A region R and the contained bends belonging to $ep^+_{S_i}(R)$ and the corresponding tunnel $tun^+_{S_i}(R)$.

Observation 8.3. *Let e be an edge with middle segments from both S_i and T_i. Then the S_i- and the T_i-endpoint of e are located in two disjoint regions of $\mathcal{R}_i$.*

Next, consider the set of S_i-endpoints $ep_{S_i}(R)$ and the set of T_i-endpoints $ep_{T_i}(R)$ inside a region $R \in \mathcal{R}_i$ independently. We partition set $ep_{S_i}(R)$ based on whether the incident segment from S_i intersects the boundary of R above or below the S_i-endpoint. We denote the corresponding sets by $ep^+_{S_i}(R)$ and $ep^-_{S_i}(R)$, respectively; see Fig. 8.5b for an illustration. Analogously, we partition set $ep_{T_i}(R)$ into sets $ep^+_{T_i}(R)$ and $ep^-_{T_i}(R)$.

By Observation 8.1, we know that most S_i- and T_i-endpoints are incident to start segments that do neither belong to S_i nor to T_i. As a consequence, segments belonging to S_i and T_i form obstacles for most start segments incident to S_i- and T_i-endpoints within a region R. Based on this notion, we are next defining regions that contain the vertices that have a visibility to most of the S_i- and T_i-endpoints inside R. For this purpose, first consider the *S_i-tunnel* $tun_{S_i}(R)$ which is the minimal region bounded by two lines parallel to segments in S_i which encloses R. Moreover, we subdivide $tun_{S_i}(R)$ into R, the part $tun^+_{S_i}(R)$ of $tun_{S_i}(R)$ that is located below R and the part $tun^-S_i(R)$ of $tun_{S_i}(R)$ that is located below R; see Fig. 8.5c. The intuition behind these tunnels is as follows: All bends in set $ep^+_{S_i}(R)$ are incident to a segment from S_i that exits R in positive y-direction. As a consequence, the part $tun^-S_i(R)$ of the tunnel $tun_{S_i}(R)$ above R is containing many obstacles that may prevent vertices in $tun^-S_i(R)$ to have a connection to one of the bends in $ep^+_{S_i}(R)$. On the other hand, the second part $tun^+S_i(R)$ may be free of obstacles and thus contain many start segments incident to bends in $ep^+_{S_i}(R)$. Analogously to $tun_{S_i}(R)$, $tun^+_{S_i}(R)$ and $tun^-_{S_i}(R)$, we define $tun_{T_i}(R)$, $tun^+_{T_i}(R)$ and $tun^-_{T_i}(R)$.

The vertices incident to bends in $ep^+_{S_i}(R)$ may have a linear distance from region R. In such a case, they may even be incident to many bends from $ep^+_{S_i}(R)$ if they are located outside of $tun^+_{S_i}(R)$. We next quantify this effect and define a cone-

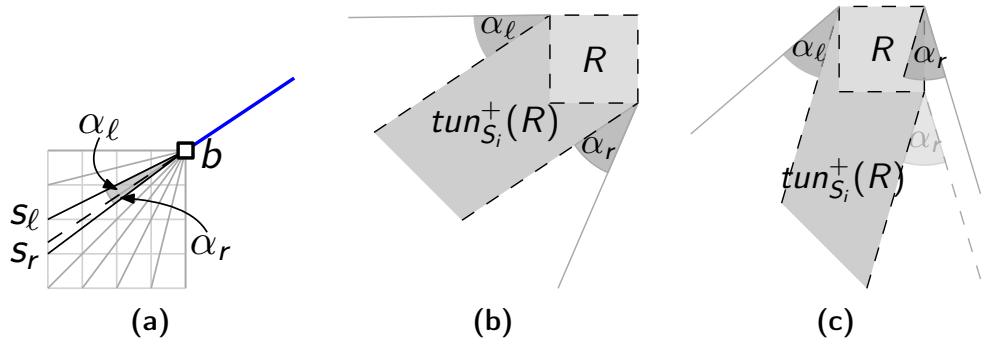


Figure 8.6: (a) The S_i^+-plausible positions for a bend $b \in ep^+_{S_i}(R)$ defined by a wedge of angle $\alpha_\ell + \alpha_r$. (b)–(c) S_i^+-plausible region $plaus^+_{S_i}(R)$ for two different slopes of segments in S_i.

shaped region of *plausible positions* in which vertices have visibility to many bends from $ep^+_{S_i}(R)$. For discretization, we consider a set of slopes $A = \{p/q | p, q \in \{-4, \ldots, 0, \ldots, 4\}\}$. Let s_ℓ and s_r denote the two slopes closest to the slope p_i/q_i of segments in S_i; see Fig. 8.6a. Moreover, let α_ℓ as well as α_r denote the angle between slopes s_ℓ and p_i/q_i as well as between s_r and p_i/q_i, respectively. It is easy to see that $0 < \alpha_\ell, \alpha_r < \pi/4$. Consider a bend $b \in ep^+_{S_i}(R)$. We call the wedge delimited by two rays of slope s_ℓ and s_r starting from b opposite of the attached segment from S_i the *S_i^+-plausible positions*; see Fig. 8.6a. We refer to the union of the S_i^+-plausible regions of all bends in $ep^+_{S_i}(R)$ as the *S_i^+-plausible region* $plaus^+_{S_i}(R)$. Note that $plaus^+_{S_i}(R)$ consists of R, $tun^+_{S_i}(R)$ and two attached wedges of angles α_ℓ and α_r attached to the left and the right side of $tun^+_{S_i}(R)$, respectively. Note that the two wedges may be attached to adjacent or opposite corners depending on the slope of segments in S_i; see Figs. 8.6b and 8.6c, respectively. Analogously, we define $plaus^-_{S_i}(R)$, $plaus^+_{T_i}(R)$ and $plaus^-_{T_i}(R)$. The following lemma asserts that indeed most start segments incident to bends in $ep^+_{S_i}(R)$ lead either to the plausible region $plaus^+_{S_i}(R)$ or to the opposite side of the tunnel $tun^-_{S_i}(R)$. We will restrict the latter case afterwards.

Lemma 8.7. *Let Γ be a RAC$_3$ drawing of a graph G with n vertices and $\Omega(n^2)$ edges in $\mathcal{O}(n^2)$ area and let $R \in \mathcal{R}_i$ be a region such that w.l.o.g. $|ep^+_{S_i}(R)| = \Omega(n^2)$. Then, vertices outside of $plaus^+_{S_i}(R) \cup tun^-_{S_i}(R)$ have a visibility to $\mathcal{O}(n)$ bends in $ep^+_{S_i}(R)$ in total.*

Proof. We first recall Observation 8.1 and the fact that only segments from T_i may intersect segments from S_i: Since at most $\mathcal{O}(n)$ start segments belong to T_i, those can be neglected for the proof of the lemma and we focus our attention on start segments that cannot intersect the segments from S_i incident to bends in $ep^+_{S_i}(R)$.

Consider a bend $b \in ep^+_{S_i}(R)$ and assume that the start segment incident to b is connected to a vertex v_b outside of $plaus^+_{S_i}(R) \cup tun^-_{S_i}(R)$. Assume w.l.o.g.

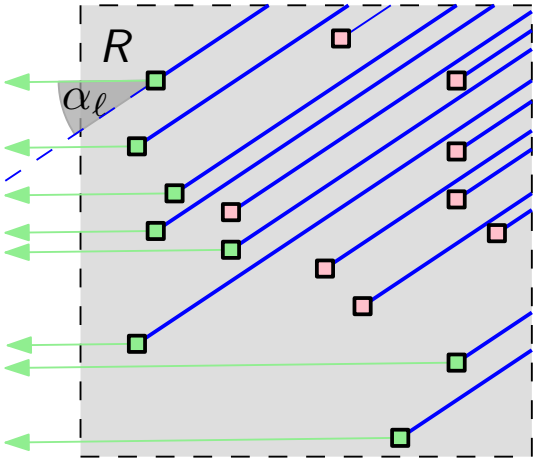


Figure 8.7: Rays of slope s_ℓ attached to the set of bends B (green) that is visible from outside $plaus^+_{S_i}(R) \cup tun^-_{S_i}(R)$. Non-visible S_i^+-endpoints are colored red.

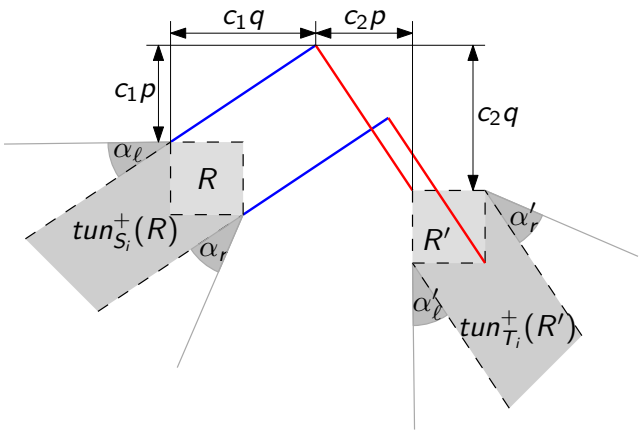


Figure 8.8: A region $R \in \mathcal{R}_i$ and one of its neighbored regions $R' \in \mathcal{N}(R)$.

that v_b is located to the left of $plaus^+_{S_i}(R) \cup tun^-_{S_i}(R)$. Observe that the slope of the start segment between b and v_b diverges by at least α_ℓ from the slope p_i/q_i of segments belonging to S_i. Thus, if b is attached to v_b, a ray of slope s_ℓ with right endpoint b is not intersecting any other segment in S_i; in particular no other segment in S_i with an endpoint in $ep^+_{S_i}(R)$.

Consider the set of S_i^+-endpoints B that admit such an intersection-free ray; see Fig. 8.7. Observe that all rays are non-overlapping and parallel. Also note that all rays hit at least one integer point inside R (the incident bend) and that the slope $s_\ell \in A$ can be expressed as $s_\ell = p_\ell/q_\ell$ for coprime integers $p_\ell, q_\ell \in \mathcal{O}(1)$. This implies that the minimum distance between two parallel rays is $\Omega(1)$. Because region R has size $\mathcal{O}(n) \times \mathcal{O}(n)$, it can only contain $\mathcal{O}(n)$ parallel rays of slope s_ℓ. We conclude that $|B| = \mathcal{O}(n)$ and the proof follows. □

Next, we consider a region $R \in \mathcal{R}_i$ and the set of its *neighbored regions*

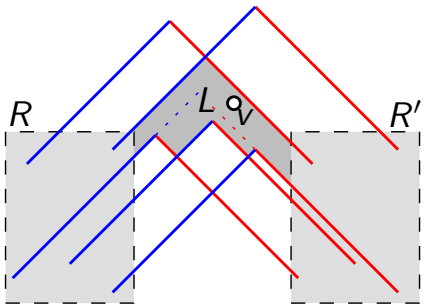


Figure 8.9: An L-tunnel L between two regions $R \in \mathcal{R}_i$ and $R' \in \mathcal{N}(R)$ that contains a vertex v.

$\mathcal{N}(R)$. Let c_1 and c_2 be two integers and let p_i/q_i be the slope of segments in S_i. Then, we say that the region R' obtained by shifting R $c_1q_i + c_2p_i$ units in x-direction and $c_1p_i - c_2q_i$ units in y-direction is a *neighbored region*; see Fig. 8.8. Intuitively speaking, region R' contains all T_i-endpoints that are part of edges that have their S_i-endpoints in R and whose middle segments are a segment from S_i of length $|c_1|\sqrt{p^2+q^2}$ and a segment from T_i of length $|c_2|\sqrt{p^2+q^2}$, respectively. Observe that the projections of R contained in $\mathcal{N}(R)$ are not necessarily contained in $\mathcal{R}_i$ while it is obviously still true that $\mathcal{N}(R) = \Theta(1)$. Also recall that by Observation 8.3, R and R' are disjoint.

We assume in the following w.l.o.g. that $c_1, c_2 > 0$. This implies that the edges with an S_i-endpoint in R and a T_i-endpoint in R' have two bends that belong to $ep^+_{S_i}(R)$ and $ep^+_{T_i}(R')$, respectively. Consider a vertex v. We say that v is an *R-vertex* if it has $\Omega(n)$ start segments incident to bends in $ep^+_{S_i}(R)$ but only $o(n)$ start segments incident to bends in $ep^+_{T_i}(R')$. Similarly, we say that v is an *R'-vertex* if it has $\Omega(n)$ start segments incident to bends in $ep^+_{T_i}(R')$ but only $o(n)$ start segments incident to bends in $ep^+_{S_i}(R)$. Observe that v can be neither R- nor R'-vertex. In the following we show that almost all edges with S_i- and T_i-endpoints in neighbored regions R and R' induce a bipartite subgraph on the two partitions defined by R- and R'-vertices. To do so, we consider edges that are not connecting an R- and an R'-vertex which we refer to as *odd edges*. The intuition behind odd edges is that these edges break the two-colorability of the induced subgraph and thus bring the graph closer to a complete subgraph. In particular, we make the following observation:

Observation 8.4. *Let e be a odd edge. Then e is incident to (i) at least one endpoint that is neither R- nor R'-vertex, or, (ii) at least one R-vertex with a start segment incident to a bend in $ep^+_{T_i}(R')$, or, (iii) at least one R'-vertex with a start segment incident to a bend in $ep^+_{S_i}(R)$.*

We refer to the special vertices discussed in Observation 8.4 as *odd endpoints*. We will first consider how odd edges may be incident to odd endpoints that are located in *L-tunnels* in the intersection of $tun^-_{S_i}(R)$ and $tun^-_{T_i}(R')$. Then, in the

proof of Lemma 8.9, we will extend the statement to the general case. Consider a vertex v in $tun^-_{S_i}(R) \cap tun^-_{T_i}(R')$. We say that v is located in an L-tunnel L if it is located inside a region bounded by edges with S_i-endpoint in R and T_i-endpoint in R' and the boundaries of R and R'; see Fig. 8.9. To be more precise, an L-tunnel is a region that contains part of $tun^-_{S_i}(R) \cap tun^-_{T_i}(R')$ and is bounded from above by a segment from S_i and a segment from T_i, from below by an alternating sequence of segments from S_i and T_i, from the left by R and from the right by R'.

Lemma 8.8. *Let Γ be a RAC_3 drawing of a graph G with n vertices and $\Omega(n^2)$ edges in $\mathcal{O}(n^2)$ area and let $R \in \mathcal{R}_i$ be a region such that w.l.o.g. $|ep^+_{S_i}(R)| = \Omega(n^2)$ and let $R' \in \mathcal{N}(R)$. Then, there are $\mathcal{O}(n)$ odd edges with a bend in $ep^+_{S_i}(R)$ and $ep^+_{T_i}(R')$ such that one of their odd endpoints is in an L-tunnel in $tun^-_{S_i}(R) \cap tun^-_{T_i}(R')$,*

Proof. First consider an odd endpoint v that is an R- or an R'-vertex: If v is the only odd endpoint for a single edge, we can ignore it as it only contributes one odd edge. Thus, there are only $\mathcal{O}(n)$ odd edges incident to endpoints with this property. Hence, we assume in the following w.l.o.g. that odd endpoints have at least two connections to $ep^+_{S_i}(R)$ and to $ep^+_{T_i}(R')$.

Assume w.l.o.g. that the slope of segments in S_i is less than 1, otherwise a symmetric argument can be applied. Then, the slope of segments in T_i is less than -1. Due to the choice of the side lengths of regions in $\mathcal{R}_i$, it follows, that the intersection $tun^-_{S_i}(R) \cap tun^-_{T_i}(R')$ is located above R'. We now show that the odd endpoints in L-tunnels in $tun^-_{S_i}(R) \cap tun^-_{T_i}(R')$ are incident to $\mathcal{O}(n)$ bends in $ep^+_{T_i}(R')$.

Consider a vertex v in an L-tunnel located in $tun^-_{S_i}(R) \cap tun^-_{T_i}(R')$. Let $B(v)$ denote the bends of $ep^+_{T_i}(R')$ that are endpoint of a start segment incident to v. We partition $B(v)$ into $B(v)^+$ and $B(v)^-$, that is, the set of bends b for which v is located in the halfplane below and above the line that is the extension of the segment from T_i incident to b, respectively.

We begin by considering two bends b and b' belonging to $B(v)$ for some vertex v. We show that the y-coordinates of b and b' differ if both b and b' belong to $B^+(v)$ or $B^-(v)$. Assume w.l.o.g. that the y-coordinate of b' is at most as large as the y-coordinate of b. First assume that b and b' belong to $B^-(v)$. Since v is located above R', the slope of the two start segments is less than -1. We observe that v can be incident to b' only if b' is located in a wedge W that is formed by the elongations of the start segment through b and of the segment from S_i through b; see Figs. 8.10a and 8.10b. If this was not the case, either the segment from T_i incident to b' would intersect the start segment incident to b as shown in Fig. 8.10a or the start segment incident to b' would intersect the segment from T_i incident to b as shown in Fig. 8.10b. We observe that the angle between the two lines spanning W is less than π which implies that no point in W has the same y-coordinate as b. We can use an analogous argument for the case where both

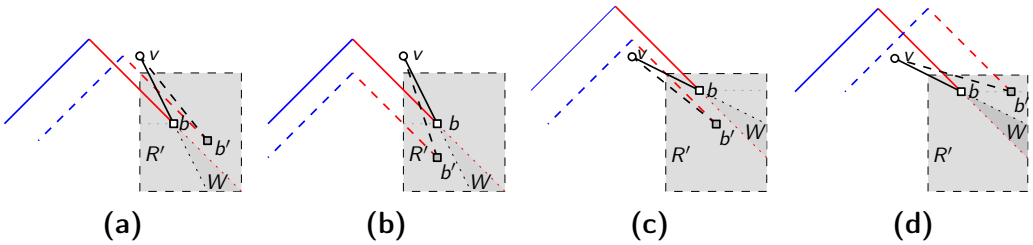


Figure 8.10: Restriction of the position of a bend b' incident to a vertex v to a wedge W under the presence of a bend b incident to v. Note that the length of segments and the size of region R' are not to scale in the illustration for better readability.

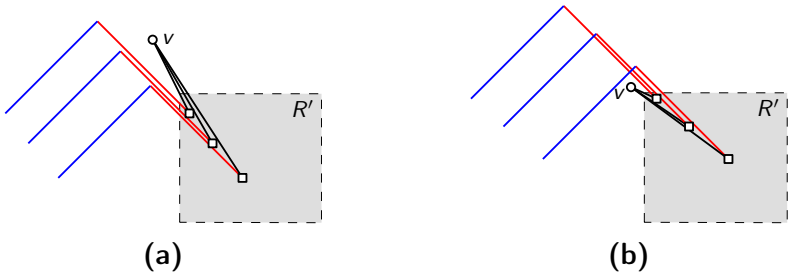


Figure 8.11: (a) Edges belonging to $B^-(v)$ do not intersect at their middle segments. (b) Edges belonging to $B^+(v)$ pairwise intersect at their middle segments. Note that the length of segments and the size of region R' are not to scale in the illustration for better readability.

bends are part of $B^+(v)$; see Figs. 8.10c and 8.10d. Note that in this case the slope of start segments is still negative but not necessarily less than one. Since all bends in $B^-(v)$ and $B^+(v)$, respectively, have distinct y-coordinates, we conclude that $B(v) = \mathcal{O}(n)$.

Since all S_i and T_i segments between R and R' have the same lengths, we observe that edges with bends in $B^-(v)$ do not intersect at their middle segments, while the edges with bends in $B^+(v)$ do so in a pairwise fashion; see Fig. 8.11. We now consider the dependencies between the bends incident to two odd endpoints v and v' in L-tunnels in $tun^-_{S_i}(R) \cap tun^-_{T_i}(R')$ where v and v' are neither R- nor R'-vertex. We will prove that the bends in $B^{\pm}(v)$ have y-coordinates distinct from those of bends in $B^{\pm}(v')$. We first consider $B^-(v)$ and $B^-(v')$. Let $b_{top}(v)$ and $b_{top}(v')$ denote the topmost bend in $B^-(v)$ and $B^-(v')$, respectively. Similarly, let $b_{bot}(v)$ and $b_{bot}(v')$ the bottommost bend in $B^-(v)$ and $B^-(v')$ respectively. We consider three cases:

C.1 v and v' appear in different L-tunnels. Assume w.l.o.g. v' is located in the half-plane below the segment from T_i through $b_{bot}(v)$. Since the start segments incident to v' have negative slope, we conclude that $b_{top}(v')$ is located below $b_{bot}(v)$; see Fig. 8.12a. Thus, the y-coordinates of bends in

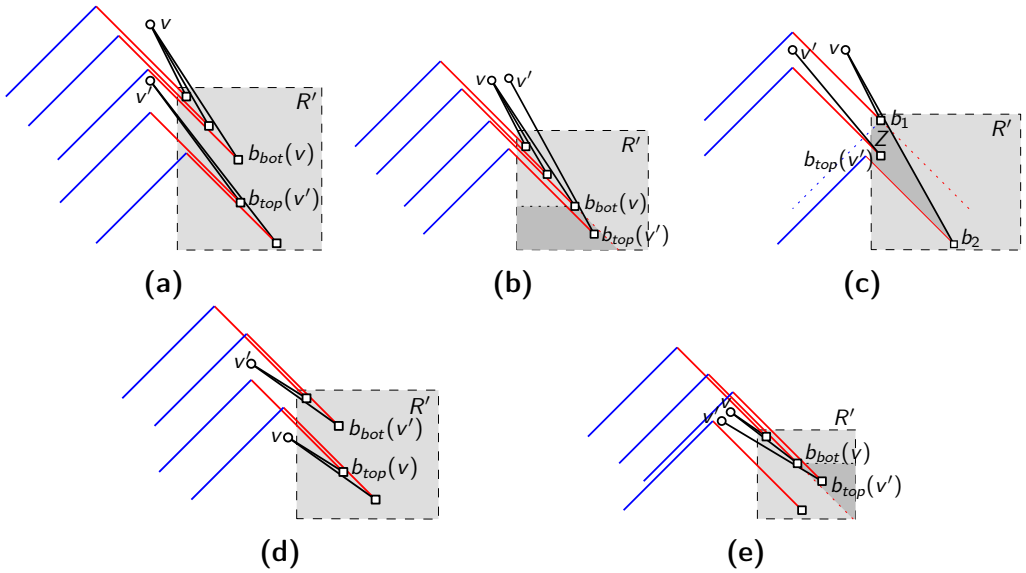


Figure 8.12: (a)–(c) Illustration for the proof that bends in $B^-(v)$ and $B^-(v')$ have distinct y-coordinates. (d)–(e) Illustration for the proof that bends in $B^+(v)$ and $B^+(v')$ have distinct y-coordinates. Note that the length of segments and the size of region R' are not to scale in the illustration for better readability.

$B^-(v)$ and $B^-(v')$ are distinct.

C.2 v and v' appear in the same L-tunnel. Assume w.l.o.g. that $b_{top}(v')$ is located in a wedge W strictly below $b_{bot}(v)$ delimited by the elongation of the segment from T_i through $b_{bot}(v)$; see Fig. 8.12b. This is true since v' does not belong to R' (and hence to W) due to the choice of the side length of R'. Thus, the y-coordinates of bends in $B^-(v)$ and $B^-(v')$ are distinct.

C.3 v' is located between the segments from T_i incident to $b_1, b_2 \in B^-(v)$. Then, the bends in $B^-(v)$ are restricted to a region Z which is bounded by two lines of the slopes of S_i and T_i that pass through b_1, a T_i-segment which is incident to b_2 and the start segment incident to b_2 as well as the boundary of R'; see Fig. 8.12c. We observe that all points in Z have smaller y-coordinate than b_1 and larger y-coordinate than b_2, that is, the y-coordinates of bends in $B^-(v)$ and $B^-(v')$ are distinct.

We point out that the line parallel to segments in S_i is part of the boundary of Z as otherwise the segment from T_i incident to a bend in $B^-(v')$ would intersect the segment from S_i belonging to the edge involving b_1. Then, the L-tunnel of v' would be intersected so that only a start segment incident to v' that belongs to T_i could connect to bends in $ep^+_{S_i}(R)$. But then, v' would be an R'-vertex that can only be incident to one bend in $ep^+_{S_i}(R)$ which is the first case we ruled out in the proof of the lemma.

The analysis for the case where $B^+(v)$ and $B^+(v')$ are considered is analogous; see Figs. 8.12d and 8.12e for the corresponding illustrations. We point out that Case C.3 is trivially covered in this scenario, since the edges involving two bends from $B^+(v)$ intersect and hence immediately make v' an R'-vertex that can only be incident to one bend in $ep^+_{S_i}(R)$.

We conclude that the y-coordinates of bends in $\bigcup_v B^-(v)$ and $\bigcup_v B^+(v)$ are distinct which implies that $|\bigcup_v B^\pm(v)| = \mathcal{O}(n)$ and hence $|\bigcup_v B(v)| = \mathcal{O}(n)$. The discussion for $ep^+_{S_i}(R)$ is analogously. □

We now summarize the partial results from Lemmas 8.7 and 8.8.

Lemma 8.9. *Let Γ be a RAC$_3$ drawing of a graph G with n vertices and $\Omega(n^2)$ edges in $\mathcal{O}(n^2)$ area and let $R \in \mathcal{R}_i$ be a region such that w.l.o.g. $|ep^+_{S_i}(R)| = \Omega(n^2)$ and let $R' \in \mathcal{N}(R)$. Then, there are $\mathcal{O}(n)$ odd edges with a bend in $ep^+_{S_i}(R)$ and $ep^+_{T_i}(R')$.*

Proof. Assume that $tun^-_{S_i}(R)$ is delimited by the elongation of two segments from S_i incident to bends in $ep^+_{S_i}(R)$. Otherwise, R can be restricted to a smaller region that fulfills the property. Further, assume for a contradiction that there are $\omega(n)$ odd edges with endpoints in $ep^+_{S_i}(R)$ and $ep^+_{T_i}(R')$. By Lemma 8.8, the odd endpoints of only $\mathcal{O}(n)$ of those odd edges are located in L-tunnels in $tun^-_{S_i}(R) \cap tun^-_{T_i}(R')$. Note that $tun^-_{S_i}(R)$ is bounded by two segments of S_i, while $tun^-_{T_i}(R')$ is bounded by two segments of T_i. Hence, all remaining odd endpoints are located outside $plaus^+_{S_i}(R) \cup tun^-_{S_i}(R)$ or outside of $plaus^+_{T_i}(R') \cup tun^-_{T_i}(R')$. According to Lemma 8.7, there are only $\mathcal{O}(n)$ bends in $ep^+_{S_i}(R)$ and $ep^+_{T_i}(R')$ that are visible from outside of $plaus^+_{S_i}(R) \cup tun^-_{S_i}(R)$ and $plaus^+_{T_i}(R') \cup tun^-_{T_i}(R')$, respectively. Thus there must be odd endpoints which are located in $plaus^+_{S_i}(R) \cap plaus^+_{T_i}(R')$. This leads to a contradiction since $plaus^+_{S_i}(R) \cap plaus^+_{T_i}(R') = \emptyset$. □

Intuitively speaking, Lemma 8.9 states that the graph induced by the edges with middle segments from S_i and T_i between regions R and R' is a bipartite graph with the exception of the $\mathcal{O}(n)$ odd edges. Therefore, the subgraph induced by edges between R and all neighbored regions $\mathcal{N}(R)$ is an almost p-partite subgraph for some constant p. Assuming that we have a drawing of K_n, this stands in contrast to the fact that the drawing is of the complete graph where all R-vertices must be pairwise connected. In the proof of the main theorem of this section, we achieve a contradiction based on these observations:

Theorem 8.2. *For sufficiently large n, there is no RAC$_3$ drawing of K_n in $\mathcal{O}(n^2)$ area.*

Proof. Assume for a contradiction that there is a RAC$_3$ drawing Γ of K_n in $\mathcal{O}(n^2)$ area for each value of $n \in \mathbb{N}$. In the following, we describe an iterative procedure that identifies a complete subgraph G' with $\Omega(n)$ vertices whose subdrawing Γ' of

Γ is drawn with $o(n^2)$ edges belonging to $E[S_i, T_i]$ for each pair of perpendicular sets of edge segments $S_i \in \mathcal{S}$ and $T_i \in \mathcal{T}$. Clearly, such a subdrawing Γ' violates Lemma 8.6 which leads to a contradiction for sufficiently large values of n.

Let c_{ST} denote the multiplicative constant from Lemma 8.6 and let $G' = (V', V' \times V')$. We compute G' iteratively and initialize it with G. We consider each of the constantly many pairs of segments $S_i \in \mathcal{S}$ and $T_i \in \mathcal{T}$ with $|E[S_i, T_i]| \geq c_{ST} n^2$ as follows.

Let p_i/q_i be the slope of segments in S_i for coprime integers p_i and q_i. Consider the checkerboard partitioning $\mathcal{R}_i$ of the drawing area into disjoint regions of side lengths $\max\{|p_i|, |q_i|\}/2$. Then, consider each of the constantly many regions $R \in \mathcal{R}_i$ and each of the neighbored regions $R' \in \mathcal{N}(R)$. Note that the number of regions R and R' over all pairs of sets S_i and T_i is a constant n_{comb}.

We iteratively perform the following procedure as long as there are at least $c_{ST}|V'|^2$ edges with a bend in $ep_{S_i}(R)$ and $ep_{T_i}(R')$ for some i, a region $R \in \mathcal{R}_i$ and a neighbored region $R' \in \mathcal{N}(R)$. Let V_R denote the set of R-vertices for region R and let $V_{R'}$ denote the set of R'-vertices for region R'. Assume w.l.o.g. that $|V_R| \geq |V_{R'}|$. By Lemma 8.9, there are only $\mathcal{O}(n)$ odd edges, say at most $c_{odd}|V'|$. Recall that vertices that are not incident to $\Omega(n)$ bends, say at least $c_R|V^*|$, in $ep_{S_i}(R)$ and $ep_{T_i}(R')$ each, are R- or R'-vertices in the graph $G^* = (V^*, E^* \times E^*)$ that is obtained at the end of our iterative procedure. Thus, there are only $2\frac{c_{odd}|V'|}{c_R|V^*|} = \mathcal{O}(1)$ vertices which are incident to a linear number of bends in $ep_{S_i}(R)$ and $ep_{T_i}(R')$ each. Thus, $|V_R| = \Omega(n)$; more precisely, $|V_R| \geq (|V'| - 2\frac{c_{odd}|V'|}{c_R|V^*|})/2$. We set G' as $(V_R, V_R \times V_R)$. Note that G' now only contains $c_{odd} n_{comb}|V| = \mathcal{O}(n) = \mathcal{O}(|V'|)$ edges that are drawn with both a bend in $ep_{S_i}(R)$ and $ep_{T_i}(R')$ since there are only n_{comb} combinations that have to be considered. Afterwards, we continue with the next iteration. Observe that in consecutive iterations, for the same combination of i, $R \in \mathcal{R}_i$ and $R' \in \mathcal{N}(R)$ in future iterations there are less than $c_{ST}|V'|^2$ edges with a bend in $ep_{S_i}(R)$ and $ep_{T_i}(R')$ for sufficiently large n as there are only n_{comb} combinations that have to be considered.

After performing all iterations of the procedure, there are less than $c_{ST}|V'|^2$ edges with a bend in $ep_{S_i}(R)$ and $ep_{T_i}(R')$ for all i, $R \in \mathcal{R}_i$ and $R' \in \mathcal{N}(R)$. Hence, Γ' contradicts Lemma 8.6 according to which $|E[S_i, T_i]| \geq c_{ST}|V'|^2$ for sufficiently large n. □

We remark that the proofs of Lemmas 8.3 and 8.7–8.9 explicitly assume quadratic area. We point out that some of the proofs cannot be directly transferred to drawings in $\mathcal{O}(n^{2+\varepsilon})$ area even for small $\varepsilon > 0$. The result from Theorem 8.1 may be regarded as a relaxation in this direction. On the other hand, the proofs of Lemmas 8.5–8.9 assume three bends per edge. In the following section, we consider RAC drawings with slightly more bends per edge to achieve quadratic area. Finally, the concept of odd edges plays a critical role in the proof of Theorem 8.2.

In Section 8.4, we will investigate the restriction to p-partite graphs which can be drawn without any odd edges.

8.3 Area Optimal RAC_8 Drawings

In this section, we show that the required area for polyline RAC drawings is $\Theta(n^2)$ by demonstrating how to draw K_n in $\mathcal{O}(n^2)$ area. The corresponding lower bound comes from the fact that the complete graph has $\Omega(n^2)$ edges. We achieve our result with just eight bends per edge, leaving only a gap of four bends per edge between our positive result and Theorem 8.2.

Theorem 8.3. *Let G be a simple graph with n vertices. Then, G admits a RAC_8 drawing in $\mathcal{O}(n^2)$ area.*

Proof. We describe how to produce a drawing of K_n in $\mathcal{O}(n^2)$ area for odd values of n; a corresponding drawing for even n is a subdrawing of the drawing of K_{n+1}. Figure 8.13 shows the drawing of K_5 obtained by our construction. The general idea of our construction is as follows: Vertices and start segments are located in the *vertex area* so that the bends of start segments can be connected to another segment with a slope s that is slightly less than 1; see Fig. 8.13b. We treat edges to be composed of two *half-edges* which are routed to the *matching area* with a sequence of segments whose slopes alternate between s and $-1/s$; see Fig. 8.13a. The matching area then contains the last bends of the two half-edges which we refer to as *matching bends* and a planar matching between those bends realized by so-called *matching segments*. A half-edge is associated with one endpoint of its corresponding edge as indicated by the coloring in Fig. 8.13, where bends of half-edges are colored in the same color as their associated endpoint. Since these segments are either parallel or perpendicular, we achieve that intersections occur at right angles. Moreover, half-edges may end in the top left half of the matching area or in the bottom right half. Finally, in the matching area, we planarly match one bend of an half-edge in the top left half with one bend of another half-edge in the bottom right half; see Fig. 8.13c.

Let $(v_0, \ldots, v_{n-1})$ be an arbitrary ordering of the vertices. Vertex v_i is placed at $(i, -i)$; see Fig. 8.13b. Consider the half-edges $\{e_i^0, \ldots, e_i^{n-2}\}$ associated with v_i and their corresponding start segments. We place the bend of the start segment of e_i^j at $(i + j + 1, j - i)$; see Fig. 8.13b. Note that all start segment bends of vertex v_i are located on a diagonal d_i of slope 1 and that the corresponding start segments are entirely located in the halfplane above d_i. Moreover, vertex v_{i-1} is located in the halfplane below d_i and thus start segments are intersection-free. We observe that the start segment bends are located within a rectangle that is rotated by $\pi/4$; see Fig. 8.13b.

The routing from vertex area to matching area is done by sequences of segments of slopes s and $-1/s$ where $s = (2n - 1)/2n$. In particular, consider the smallest

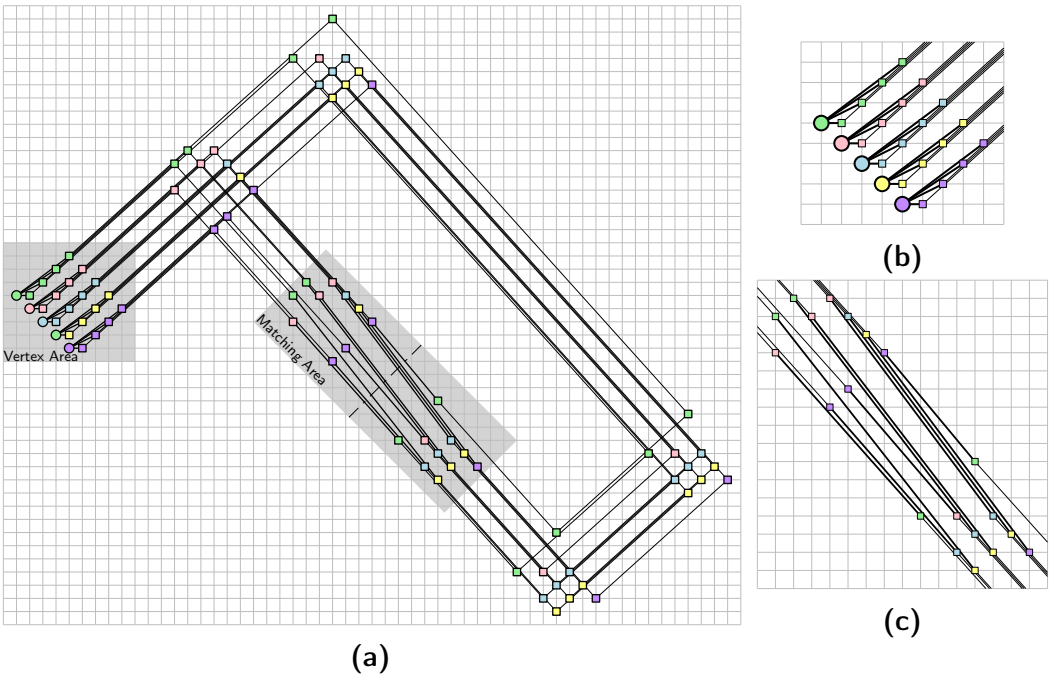


Figure 8.13: A RAC$_3$ drawing of K_5 produced by the algorithm in the proof of Theorem 8.3 in 55×47 area. Vertices are drawn as circles and bends as squares. The bends of half-edges are colored according to their associated endpoint. (a) Overview of the drawing, (b) zoom into vertex area, (c) zoom into matching area.

rectangle R rotated by $\pi/4$ that contains all start segment bends. By attaching a segment of slope s to each bend in R, the next integer points used by these segments are located outside of R. This procedure allows to create "copies" of R and its contained bends at $k \cdot 2n$ horizontal and $k \cdot (2n-1)$ vertical distance for some $k \in \mathbb{N}$. The same can be done with segments of slope $-1/s$. In addition, recall that the bends of start segments of v_i are located on diagonal d_i of slope 1. Since s is slightly less than 1, the new segments do not intersect start segments; see Fig. 8.13b. Similarly, since $-1/s$ is slightly less than -1, the bends in the top left half of the matching area are accessible from the bottom right without intersections, while the bends in the bottom right half are accessible from the top left; see Fig. 8.13c. The bends of edges are defined more precisely as follows. Consider half-edge e_i^j. Recall that the first bend is at $(i+j+1, j-i)$. If e_i^j is routed to the top left half of the matching area, the remaining bends are at $(2n+i+j+1, 2n+j-i-1)$ and $(4n+i+j, j-i-1)$. Otherwise, e_i^j is routed to the bottom right half of the matching area and its remaining bends are at $(4n+i+j+1, 4n+j-i-2)$, $(10n+i+j-2, -2n+j-i-2)$, $(8n+i+j-2, -4n+j-i-1)$ and $(6n+i+j, -2n+j-i-1)$. Note that $(4n+i+j, j-i-1)$ is the position of the potential bend in the top left half of the matching area whereas $(6n+i+j, -2n+j-i-1)$ is the position of the potential bend in the bottom right half. In the following we refer to these bends as *top-left*

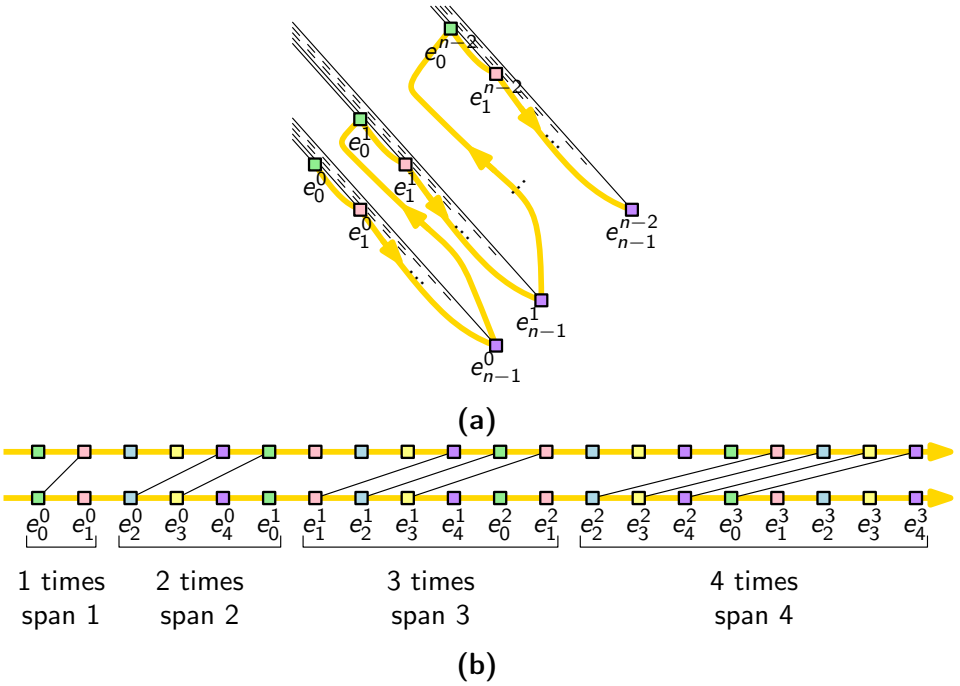


Figure 8.14: Matching between matching bends in the matching region. (a) Accessability of matching bends, and, (b) matching assignment for $n = 5$ as it is used in Fig. 8.13.

and *bottom-right matching bends*, respectively.

Before we discuss how to connect bends in the matching area intersection-free such that all pairs of vertices are adjacent, we discuss the area. The leftmost x-coordinate is 0 for vertex v_0 while the rightmost x-coordinate is $12n - 5$ for a bend of e_{n-1}^{n-2}. The bottommost y-coordinate is $-5n$ for a bend of e_{n-1}^{0} while the topmost y-coordinate is $5n - 4$ for a bend of e_0^{n-2}. Thus the drawing fits in area $(12n - 5) \times (10n - 3)$.

It remains to discuss how to connect matching bends with so-called *matching segments*. To this end, first consider how the matching bends are accessible from the opposite half of the matching area. The first accessible matching bend belongs to half-edge e_0^0, followed by a matching bend belonging to e_1^0. This pattern repeats until all e_i^0 are encountered, in increasing order of i; see Fig. 8.14a. Note that each of those matching bends is connected to a different vertex as indicated by the coloring in Fig. 8.14a. After all e_i^0, the matching bends of half-edges all e_i^1 are encountered, again in increasing order of i. This pattern repeats for half-edges e_i^j in increasing order of j until all matching bends have been encountered; see Fig. 8.14a.

Based on the ordering defined by the accessibility of matching bends, we define a matching assignment between top-left and bottom-right matching bends. First, observe that the orderings of top-left and bottom-right matching bends are

identical. To define the matching, we make use of the notion of a span. The *span* of an edge is equal to the distance of its endpoints in a total ordering. For the matching, we first connect the first bottom right matching bend with the second top left matching bend. This edge has span one. Then, we connect the next two bottom right matching bends with the subsequent two top-left matching bends. The resulting two edges have span two. We repeat this pattern, creating k edges of span k for all values $1 \leq k \leq n-1$; see Fig. 8.14b for the case where $n = 5$.

In the following, we show that this matching creates a connection between every pair of vertices. The matching contains exactly k edges of span k for all values of $1 \leq k \leq n-1$. Moreover, for $k \leq (n-1)/2$, we observe that edges of span k and span $n-k$ behave similarly: A segment of span k connects the bottom right matching bend incident to vertex v_i to the top left matching bend of vertex $v_{(i+k) \bmod n}$. On the other hand, a segment of span $n-k$ connects the top left matching bend of vertex v_i with the bottom right matching bend of vertex $v_{(i-(n-k)) \bmod n} = v_{(i+k) \bmod n}$. Hence, it remains to show that the bottom right matching bends incident to matching segments of span k are connected to different vertices than the top left matching bends incident to matching bends of span $n-k$ for all $1 \leq k \leq (n-1)/2$. For this, we apply an inductive argument. Clearly, the claim is true when $k = 1$. Now assume, that for k all such matching bends are connected to distinct vertices. Let v_i be the vertex connected to the first bottom right matching bend incident to matching segments of span k and let v_j be the vertex connected to the first top left matching bend incident to matching segments of span $n-k$. Clearly, the last bottom right matching bend incident to matching segments of span k is incident to vertex $v_{(i+k) \bmod n}$. Conversely, the last bottom right matching bend incident to matching segments of span k is incident to vertex $v_{(j+n-k) \bmod n} = v_{(j-k) \bmod n}$. Since all matching bends are incident to distinct vertices, it follows that $j = (i+1) \mod n$. Now consider the matching segments of span $k+1$ and $n-k-1$. The bottom right matching bends incident to edges of span k are connected to vertices $v_{(i+1) \bmod n}, \ldots, v_{(i+k+2) \bmod n}$ while the top left matching bends incident to matching segments of span $n-k-1$ are connected to vertices $v_{(j-1) \bmod n}, \ldots, v_{(j-1-(n-k-1)) \bmod n} = v_{(j-1) \bmod n}, \ldots, v_{(j+k-n) \bmod n} = v_{(i) \bmod n}, \ldots, v_{(i-(n-k-1)) \bmod n}$. Thus, all connected vertices are distinct as required.

It remains to show that the (straight-line) matching segments are indeed planar. First note that the span of matching segments is bounded by $n-1$. Hence, the bottom right matching bend of half-edge e_i^j will be matched with a top left matching bend of some half-edge e_k^ℓ such that $\ell \in \{j, j+1\}$. Let d_j^{tl} denote the diagonal of slope -1 passing through the top left matching bends of half-edges e_i^j and let d_j^{br} denote the diagonal of slope -1 passing through the bottom right matching bends of half-edges e_i^j. Observe that d_j^{tl} is located in between d_j^{br} and d_{j+1}^{br} such that the distance to both d_j^{br} and d_{j+1}^{br} is the same; see Fig. 8.15. In order to prove that matching segments are planar, we show that the line through

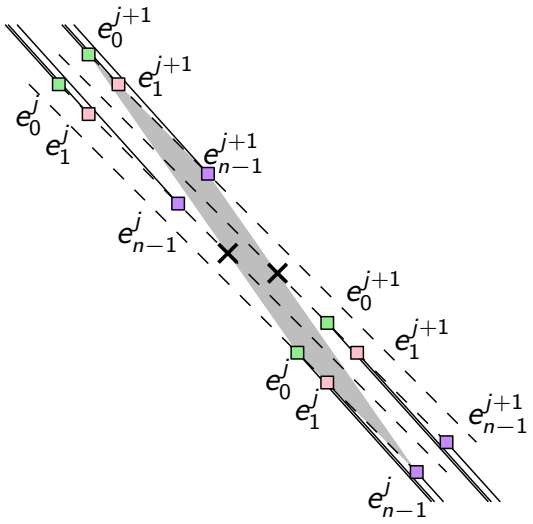


Figure 8.15: A matching segment connecting bends of half-edges e_i^j and e_k^{j+1} occurs in the gray shaded area.

the top left matching bend of e_0^{j+1} and through the bottom right matching bend of e_0^j intersects d_j^{tl} to the right of the top left matching bend of e_{n-1}^j; see crosses in Fig. 8.15. Note that a symmetric argument holds for the intersection of the line through the top left matching bend of e_{n-1}^{j+1} and through the bottom right matching bend of e_{n-1}^j with d_{j+1}^{br} and the bottom right matching bend of e_0^{j+1}.

Recall that the top left matching bend of e_0^{j+1} is located at $(4n+j+1, j)$ while the bottom right matching bend of e_0^j is located at $(6n+j-1, -2n+j-1)$. Hence the line through both bends has slope $-(2n+1)/(2n-2)$. It is possible to compute its line equation:

$$y = \frac{2n+1}{2n-2}x + \frac{8n^2+4nj+6n-j+1}{2n-2}. \tag{8.1}$$

In addition, d_j^{tl} has slope -1 and passes through the top left matching bend of e_{n-1}^j located at $(5n+j-1, -n+j)$. We can compute its line equation:

$$y = -x + 4n + j + 1. \tag{8.2}$$

With Eqs. (8.1) and (8.2), we can compute the x-coordinate of the intersection point of both lines which is $x = \frac{16}{3}n + j - \frac{1}{3}$. This is to the right of e_{n-1}^j which is located at $x = 5n + j - 1$ as claimed. We conclude that the drawing is indeed a RAC$_8$ drawing because there are three bends on half-edges routed to the top left half of the matching area and five bends on half-edges routed to the bottom right half. □

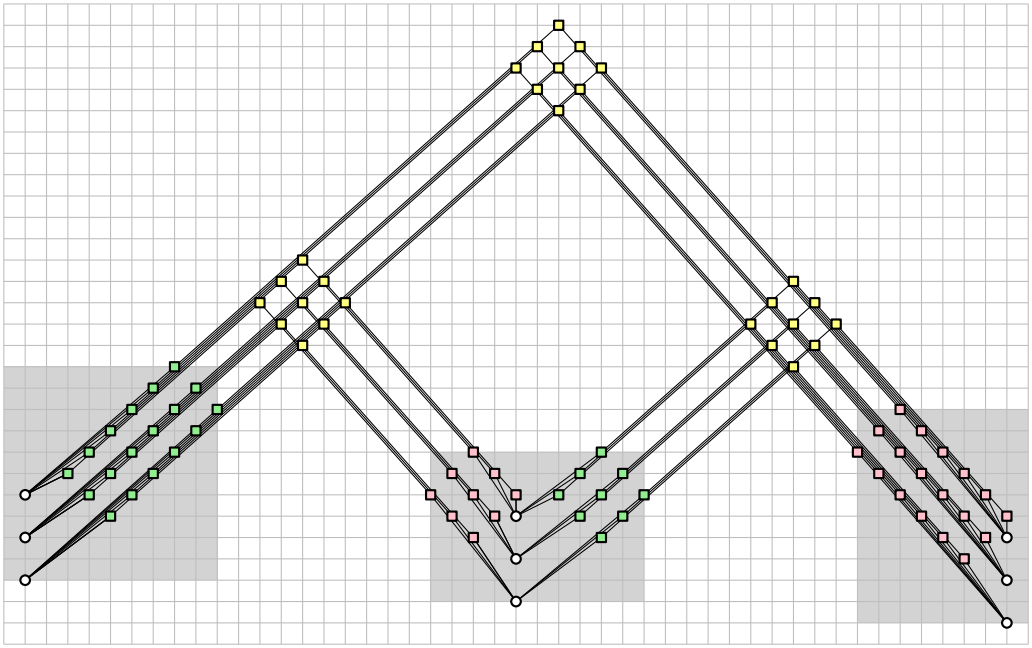


Figure 8.16: A RAC$_3$ drawing of $K_{3,3,3}$ produced by the algorithm in the proof of Theorem 8.4 in 46×28 area. Vertices are drawn as circles, bends of edges as squares.

8.4 Area Optimal RAC$_3$ Drawings of p-partite Graphs

In this section, we shift our attention to the class of p-partite graphs, that is, the graphs whose vertices can be partitioned into p sets such that there exists no edge between a pair of vertices from the same set. In the restricted setting where $p = \mathcal{O}(1)$, we will improve upon our results from Section 8.1 and Section 8.3 and show how to compute a RAC$_3$ drawing in $\mathcal{O}(n^2)$ area. We also point out that the class of p-partite graphs is a natural candidate for such an improvement since our negative result in Theorem 8.2 relied on finding a suitably large clique in the graph that cannot be realized while a clique in a p-partite graph consists of at most p vertices.

Theorem 8.4. *Let G be a simple p-partite graph with n vertices. Then, G admits a RAC$_3$ drawing in $\mathcal{O}(n^2p^4)$ area.*

Proof. We describe a drawing for $K_{(n_p)_p}$, that is, the complete p-partite graph with n_p vertices in each partition. If G has partitions of different sizes, we augment G to $K_{(n_p)_p}$ where n_p is the number of vertices in the largest partition of G. Then, it holds that $n_p < n$ and $pn_p \geq n$. Figure 8.16 shows an example drawing of $K_{3,3,3}$ obtained by our construction that we will discuss in the following.

We enumerate the partitions arbitrarily from 0 to $p-1$. Let $(v_0^j, \ldots, v_{n_p-1}^j)$ be an arbitrary ordering of the vertices of partition j with $0 \leq j \leq p-1$. Vertex v_i^j is positioned at $(2pn_pj + 2n_pj - j, 2i - j)$; see circles in Fig. 8.16. Consider an edge $e = (v_i^j, v_k^\ell)$ such that $j < \ell$. We draw e as a sequence of segments with the following three bends:

(i) The bend directly incident to v_i^j is at $(2pn_pj + n_p\ell + n_pj + k - i - j + 1, n_p\ell - n_pj + i + k - j)$; see green squares in Fig. 8.16. We also refer to this bend as a *right bend* of v_i^j.

(ii) The middle bend is at $(pn_p\ell + pn_pj + n_p\ell + n_pj + k - i - j + 1, pn_p\ell - pn_pj + n_p\ell - n_pj + i + k - \ell)$; see yellow squares in Fig. 8.16.

(iii) The bend directly incident to v_k^ℓ is at $(2pn_p\ell + n_p\ell + n_pj + k - i - \ell + 1, n_p\ell - n_pj + i + k - \ell))$; see red squares in Fig. 8.16. We also refer to this bend as a *left bend* of v_k^ℓ.

Clearly, the leftmost x-coordinate assigned is 0 for vertices of partition 0, while the rightmost x coordinate is $p^2n_p - 2n_p - p + 1$ for vertices of partition $p-1$. On the other hand, the bottommost assigned y-coordinate is $-p+1$ for vertex v_0^{p-1} while the topmost assigned y-coordinate is $p^2n_p + n_p - p - 1$ for the middle bend of edge $(v_{n_p-1}^0, v_{n_p-1}^{p-1})$. Since as discussed before, $n_p < n$, it follows that the total required area is $\mathcal{O}(n^2p^4)$.

Hence, it remains to discuss that the drawing is indeed a RAC drawing. We first observe that all middle segments have slope $(pn_p-1)/pn_p$ or slope $-pn_p/(pn_p-1)$; see the segments between green and yellow or yellow and red bends, respectively. We conclude that all proper intersections of middle segments are at right angles. In the following, we show that these intersections are indeed the only intersections.

We start by considering the start regions of two partitions j and $\ell > j$; see the gray shaded areas in Fig. 8.16. The rightmost start segment bend incident to a vertex in partition j has x-coordinate $2pn_pj + pn_p + n_pj - j$ and belongs to edge $(v_0^j, v_{n_p-1}^{p-1})$. On the other hand, the leftmost start segment bend incident to a vertex in partition ℓ belongs to edge $(v_{n_p-1}^0, v_0^\ell)$ and has x-coordinate $2pn_p\ell + n_p\ell - n_p - \ell + 2 \geq 2pn_pj + 2pn_p + n_pj - j + 1$ since $\ell \geq j+1$. Thus, the leftmost x-coordinate of a start segment bend incident to a vertex in partition ℓ is at least $pn_p + 1$ larger than the rightmost x-coordinate of a start segment bend incident to a vertex in partition j and start regions from different partitions are disjoint.

We now show that the start segments incident to vertex v_i^j are intersection-free. We first observe that all right bends of v_i^j are located on a common diagonal $d_r(v_i^j)$ of slope 1 while the left bends of v_i^j are located on a common diagonal $d_\ell(v_i^j)$ of slope -1.; see the green and red bends in Fig. 8.17, respectively. By construction, vertex v_i^j is located on the intersection of the diagonal $d'_\ell(v_i^j)$ one unit below $d_\ell(v_i^j)$ and of the diagonal $d'_r(v_i^j)$ one unit above $d_r(v_i^j)$; see dashed-dotted diagonals in

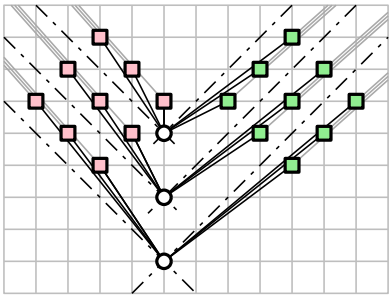


Figure 8.17: Detail of the start area of the middle partition in Fig. 8.16.

Fig. 8.17. As a result the start segments connecting v_i^j to its right bends are located in the half plane above $d_r(v_i^j)$, while the start segments connecting v_i^j to its left bends are located in the half plane below $d_\ell(v_i^j)$. In addition, v_i^j has a vertical distance of at least two to vertex v_k^j with $k \neq i$ in the same partition. Therefore, their incident start segments do not intersect each other; see Fig. 8.17. Finally, the right bends of v_i^j are connected to a middle segment of slope slightly less than 1. Similarly, the left bends of v_i^j are connected to a middle segment of slope slightly less -1. Hence, there is no intersection between the start segments of v_i^j and these middle segments; see Fig. 8.17. Moreover, since the slope of middle segments incident to right and left bends is almost 1 and -1, respectively, they also will only intersect the diagonals $d'_r(v_k^j)$ and $d'_\ell(v_k^j)$ of another vertex v_k^j at least $pn_p - 1$ units to the right and left, respectively. Moreover diagonals $d_r(v_k^j)$ and $d_\ell(v_k^j)$ of v_k^j are intersected at least $2(pn_p - 1)$ units to the right and left, respectively, and thus the middle segments incident to start segment bends of v_i^j will not intersect the start segments of v_k^j.

It remains to show that start segments of a partition j cannot be intersected by a middle segment incident to a start segment bend of partition $\ell \neq j$. We show this statement by proving that in the range of the x-coordinates covered by the start region of partition j, all middle segments incident to a start segment bend of partition ℓ are above the start region of partition j. The topmost y-coordinate of a right bend in j occurs on edge $(v^j_{n_p-1}, v^p_{n_p-1})$ and is equal to $pn_p - n_pj + 2n_p - j - 2$. Conversely, the topmost y-coordinate of a left bend in j occurs on edge $(v^0_{n_p-1}, v^j_{n_p-1})$ and is equal to $n_pj + 2n_p - j - 2$. We conclude that the topmost y-coordinate of any start segment bend is at most $pn_p + 2n_p - j - 2$.

First, consider the case where $\ell > j$. More precisely, we assume that $\ell = j + 1$ since the middle segments of partitions $\ell' > \ell$ are located above those of partition ℓ. The horizontal distance between partitions j and ℓ is at least $pn_p + 1$. At the leftmost x-coordinate belonging to the start region of partition ℓ, we encounter the left bend of edge $(v^0_{n_p-1}, v^\ell_0)$ which has y-coordinate $n_p\ell + n_p - \ell - 1 = n_pj + 2n_p - j - 2$. Conversely, if we continue k units in positive x-direction, we have a minimum distance of $pn_p + 1 + k$ towards partition j and encounter a left bend with y-coordinate $n_pj + 2n_p - j - 2 - k$. Since the slope of middle

segments incident to left bends is $-pn_p/(pn_p-1)$, we conclude that any such middle segment has y-coordinate at least $(pn_p+1+k)\cdot\frac{pn_p}{pn_p-1}+n_pj+2n_p-j-2-k > pn_p+k+n_pj+2n_p-j-2-k = pn_p+n_pj+2n_p-j-2$ which is larger than pn_p+2n_p-j-2.

Second, consider the case where $\ell < j$. More precisely, we assume that $\ell = j-1$ since the middle segments of partitions $\ell' < \ell$ are located above those of partition ℓ. The horizontal distance between partitions j and ℓ is at least pn_p+1. At the rightmost x-coordinate belonging to the start region of partition ℓ, we encounter the right bend of edge $(v_0^\ell, v_{n_p-1}^{p-1})$ which has y-coordinate $pn_p-n_p\ell+n_p-\ell-2 = pn_p-n_pj+2n_p-j-1$. Conversely, if we continue k units in negative x-direction, we have a minimum distance of pn_p+1+k towards partition j and encounter a left bend with y-coordinate $pn_p-n_pj+2n_p-j-1-k$. Since the slope of middle segments incident to left bends is $(pn_p-1)/pn_p$, we conclude that any such middle segment has y-coordinate at least $(pn_p+1+k)\cdot\frac{pn_p-1}{pn_p}+pn_p-n_pj+2n_p-j-1-k = 2pn_p-n_pj+2n_p-j-2+-k/pn_p$. Since $k < jn_p$, we have that $2pn_p-n_pj+2n_p-j-2+-k/pn_p > 2pn_p-n_pj+2n_p-j-2+-j/p$. In addition, $j \leq (p-1)$ and we conclude that $2pn_p-n_pj+2n_p-j-2+-j/p > pn_p+3n_p-j-2-(p-1)(p)$ which is larger than pn_p+2n_p-j-2.

Finally, we conclude that no two middle segments overlap. Middle segments incident to start segment bends of the same partition do not overlap by construction, while we showed that middle segments do not pass through start regions of other partitions which would be the case if two middle segments of two different partitions overlapped. □

We remark that the class of k-planar graphs is a subclass of the $\Theta(\sqrt{k})$-partite graphs [137]. Using Theorem 8.4, we obtain the following result for k-planar graphs:

Corollary 8.1. *Let G be a k-planar graph with n vertices. Then, G admits a (not necessarily k-planar) RAC_3 drawing in $\mathcal{O}(n^2k^2)$ area.*

In particular, note that when applying the construction in the proof of Theorem 8.4, the resulting drawing may be $\Omega(n)$-planar even if the input is a k-planar graph. For instance, if two vertices in the same color class have degree $\Omega(n)$, their incident edges will form $\Omega(n^2)$ intersections.

Chapter 9

Conclusions

In this thesis, we made contributions to the literature that discusses several drawing styles which extend beyond the traditional principles employed in graph drawing. The results presented in this thesis may be regarded as typical for present day graph drawing research. In this chapter, we summarize our results and state some related open problems.

Beyond orthogonal drawings. In Part I, we considered two extensions of the widely used orthogonal graph drawing model: smooth orthogonal and octilinear drawings. Morever, we initiated the study of orthogonal drawings beyond planarity.

More precisely, in Chapter 3, we continued the study on smooth orthogonal and octilinear drawings of planar graphs. We showed that the class of graphs admitting a planar smooth orthogonal drawing of curve complexity one and the class of graphs admitting a planar octilinear drawing of curve complexity one are incomparable. With previous results, this classifies the relationships between the classes of graphs admitting planar drawings in either model. We also proved that it is NP-hard to decide whether a given smooth orthogonal or octilinear representation of a graph of maximum degree four admits a realizable drawing. Curiously, this problem is analogous to the Metrics step in the Topology-Shape-Metrics framework of orthogonal graph drawing which is known to be polynomial time-solvable. Here, an interesting open problem is to study, whether a similar result holds when the input is an embedded graph instead of a representation. We conjecture that NP-hardness still holds in this setting, however, it appears to be difficult to find suitably rigid gadgets for a reduction as the maximum degree is four. Finally, we provided algorithms for computing bi-monotone smooth orthogonal and octilinear Kandinsky drawings with curve complexity two for triangulations. In particular, in the smooth orthogonal model, we also tried to ensure that many edges are drawn with one single segment. Generalizing this result for non-triangulated planar graphs and improving the aspect ratio and how many edges can be drawn with a single segment are future research directions to obtain high quality graph drawings that

can be used in practical applications. Moreover, it may be worthwhile to verify the presumably good readability of smooth orthogonal drawings in a user study.

In Chapter 4, we extended the study on orthogonal and smooth orthogonal drawings to 1-planar graphs of maximum degree four. We studied both the general 1-planar and outer-1-planar setting and proved upper and lower bounds for the required curve complexity of embedding preserving drawings. For the general 1-planar setting we showed that there is always an OC_4-drawing and an SC_3-drawing in quadratic and cubic area, respectively. In the outer-1-planar setting we showed that OC_3-drawings and SC_2-drawings are always achievable in quadratic and exponential area, respectively. We also provided lower bound constructions that showed that with one exception all of our curve complexity results are tight; we leave the question whether SC_3 is required for some embedded 1-planar graph as an open problem. Moreover, most of our upper bound results assumed biconnectivity. While it seems likely that they can be generalized to the simply connected case, we leave this as a future research direction. Another natural question arising from our results is whether outer-1-planar graphs admit an embedding preserving SC_2-layout in polynomial area. Moreover, while we already investigated outer-1-planar graphs, other meaningful subclasses like triconnected 1-planar or IC-planar graphs are of interest. Finally, the main open problem posed by this chapter is whether similar results can be achieved for other beyond planar graph classes such as 2-planar graphs. In particular, such embedding preserving algorithms for more complex beyond planar graph classes may prove useful in several applications.

Beyond stack layouts. In Part II, we considered two graph drawing models closely related to stack layouts of graphs, namely, queue layouts and arc diagrams. While many results exist for stack layouts of planar graphs in the literature, we made significant progress in understanding queue layouts for this graph class. We also investigated down-up monotone arc diagrams which can be seen as an extension of 2-page stack layouts.

More precisely, in Chapter 5, we proved that the queue number of the class of planar graphs of bounded degree is bounded by a constant. This result partially answers a long-standing conjecture by Heath, Leighton and Rosenberg and also stands in contrast to general bounded degree graphs for which the queue number cannot be bounded by a constant. Moreover, we showed how to compute a queue layout achieving our bound in polynomial time. This result has interesting implications, most notably, it implies that bounded degree planar graphs admit 3D straight-line drawings in linear volume. In the meanwhile, the most natural open problem has been settled: Namely, it was shown that the queue number of all planar graphs is at most 49 [76]. The gap to the best known lower bound of 4 for the queue number of planar graphs [8] is however still significant. Most likely, new concepts are required for closing the gap. As an intermediate step, it may be worthwhile to investigate so-called *mixed linear layouts* in which the edges assigned

to some pages do not nest (*queue pages*) while the edges assigned to some other pages do not intersect (*stack pages*). Recently, it was shown that not all planar graphs admit a mixed layout with one stack and one queue page [139]. It may be worthwhile to try to achieve a mixed layout with at most three stack pages and less than 49 queue pages for every planar graph. From a practical point of view, queue layouts are especially important in 3D graph drawing. In this context, however, queue layouts are used to create track layouts which in turn are converted to the 3D graph drawing. Hence, efficient layout algorithms that directly produce track layouts may improve the quality of the resulting drawing.

In Chapter 6, we considered down-up monotone arc diagrams of planar graphs. We showed that in this setting $15/16n-\mathcal{O}(1)$ biarcs are sufficient, which is the first upper bound of form $c \cdot n-\mathcal{O}(1)$ for $c < 1$. The analysis of our algorithm relied on amortized analysis and in fact only few cases were tight regarding the allocation of credits. Hence, it seems likely that a further refinement is possible to improve upon our result. In addition, we gave a SAT formulation for computing arc diagrams with a given upper bound of biarcs. While we performed some experiments with our implementation mainly focusing on Kleetopes which are graphs with many separating triangles, it remains open whether there exists a graph that requires strictly more biarcs in any of its monotone arc diagrams compared to its non-monotone arc diagram with the fewest biarcs. Nevertheless, our SAT formulation could be a useful tool in narrowing the gap between upper and lower bound.

Beyond planar drawings. In Part III, we considered polyline RAC drawings which are a well-known type of beyond planar drawings. Specifically, we made a contribution towards closing the few remaining gaps on the density in the literature and we presented new results on the required area of RAC drawings of dense graphs.

More precisely, in Chapter 7, we showed that the maximum edge density of RAC_1 graphs is $5.5n - \mathcal{O}(1)$ which is tight up to an additive constant. We leave finding the correct additive constant as an open problem; in particular, our lower bound construction suggests that some of the 3-planar graphs of maximum edge density may be RAC_1 graphs. In addition, we proved that the class of simple RAC_1 graphs, that is graphs that admit a simple RAC_1 drawing, has a maximum edge density of at most $5.4n - 10.8$ while we also gave a lower bound construction with $5n - 10$ edges. Narrowing this gap is an interesting research question and we conjecture that indeed the maximum edge density of simple RAC_1 graphs is $5n-10$. A starting point for this line of research could be the observation that the planarization of a simple RAC_1 graph does not contain any lenses which play an important role in the proof of the upper bound. The maximum edge density of RAC_2 graphs is another open problem; the best known upper and lower bounds in the literature have a gap of approximately $67n$. We also point out two interesting properties of our lower bound constructions: First, the area required by the drawings in our lower bound proofs is exponential and hence it may be worthwhile to investigate

whether the same density results hold for RAC_1 graphs that admit RAC_1 drawings in polynomial area. Second, our two lower bound constructions are closely related to the known 2- and 3-planar graphs of maximum edge density. Thus, investigating the relationship between RAC_1 and 3-planar graphs as well as between simple RAC_1 and 2-planar graphs may be of interest. Finally, for both RAC_1 and RAC_2 graphs the characterization and recognition problems are not settled yet; especially the recognition could also prove useful in practical applications.

In Chapter 8, we investigated the area requirement of RAC drawings of general graphs. We improved the best known upper bound for the area of RAC_3 drawings from $\mathcal{O}(n^4)$ to $\mathcal{O}(n^3)$ area. At the same time, this result improves upon a known algorithm which achieves RAC_4 drawings in $\mathcal{O}(n^3)$ area. We complemented this result by showing that K_n does not admit a RAC_3 drawing in $\mathcal{O}(n^2)$ area for sufficiently large n. A future direction of research will be to close this gap, we believe that our lower bound proof may be generalized to yield a better bound. We then showed that in two relaxations quadratic area can be achieved. Namely, every graph admits a RAC_8 drawing in $\mathcal{O}(n^2)$ area, while for p-partite graphs with constant p, even a RAC_3 drawing is possible. We state the question how many bends are sufficient to guarantee the existence of a RAC drawing in quadratic area as an open question. Moreover, our upper bound constructions result in non-simple drawings. Thus, it may be worthwhile to consider also the area requirement of simple RAC drawings. Finally, we also emphasize that our algorithms should rather be regarded as proofs of concept. For practical applications, algorithms must take into account other parameters next to area and curve complexity to provide easy-to-read quadratic area RAC drawings.

Bibliography

[1] B. M. Ábrego, O. Aichholzer, S. Fernández-Merchant, P. Ramos, and G. Salazar. Shellable drawings and the cylindrical crossing number of K_n. *Discret. Comput. Geom.*, 52(4):743–753, 2014.

[2] E. Ackerman. On the maximum number of edges in topological graphs with no four pairwise crossing edges. *Discret. Comput. Geom.*, 41(3):365–375, 2009.

[3] E. Ackerman. On topological graphs with at most four crossings per edge. *Comput. Geom.*, 85, 2019.

[4] E. Ackerman, B. Keszegh, and M. Vizer. On the size of planarly connected crossing graphs. *J. Graph Algorithms Appl.*, 22(1):11–22, 2018.

[5] E. Ackerman and G. Tardos. On the maximum number of edges in quasi-planar graphs. *J. Comb. Theory, Ser. A*, 114(3):563–571, 2007.

[6] P. K. Agarwal, B. Aronov, J. Pach, R. Pollack, and M. Sharir. Quasi-planar graphs have a linear number of edges. *Combinatorica*, 17(1):1–9, 1997.

[7] M. Ajtai, V. Chvátal, M. Newborn, and E. Szemerédi. Crossing-free subgraphs. *North-Holland Mathematics Studies*, 60(C):9–12, Jan. 1982.

[8] J. M. Alam, M. A. Bekos, M. Gronemann, M. Kaufmann, and S. Pupyrev. Queue layouts of planar 3-trees. In T. C. Biedl and A. Kerren, editors, *Graph Drawing and Network Visualization - 26th International Symposium, GD 2018, Barcelona, Spain, September 26-28, 2018, Proceedings*, volume 11282 of *Lecture Notes in Computer Science*, pages 213–226. Springer, 2018.

[9] M. J. Alam, M. A. Bekos, M. Kaufmann, P. Kindermann, S. G. Kobourov, and A. Wolff. Smooth orthogonal drawings of planar graphs. In A. Pardo and A. Viola, editors, *LATIN 2014: Theoretical Informatics - 11th Latin American Symposium, Montevideo, Uruguay, March 31 - April 4, 2014. Proceedings*, volume 8392 of *Lecture Notes in Computer Science*, pages 144–155. Springer, 2014.

[10] M. J. Alam, F. J. Brandenburg, and S. G. Kobourov. Straight-line grid drawings of 3-connected 1-planar graphs. In S. K. Wismath and A. Wolff, editors, *Graph Drawing - 21st International Symposium, GD 2013, Bordeaux, France, September 23-25, 2013, Revised Selected Papers*, volume 8242 of *Lecture Notes in Computer Science*, pages 83–94. Springer, 2013.

[11] P. Angelini, M. A. Bekos, H. Förster, and M. Gronemann. Bitonic st-orderings for upward planar graphs: The variable embedding setting. In *Proceedings of the 46th International Workshop on Graph-Theoretic Concepts in Computer Science*, 2020. To be published.

[12] P. Angelini, M. A. Bekos, H. Förster, and M. Kaufmann. On RAC drawings of graphs with one bend per edge. *Theoretical Computer Science*, 2020.

[13] P. Angelini, M. A. Bekos, M. Kaufmann, M. Pfister, and T. Ueckerdt. Beyond-planarity: Turán-type results for non-planar bipartite graphs. In W. Hsu, D. Lee, and C. Liao, editors, *29th International Symposium on Algorithms and Computation, ISAAC 2018, December 16-19, 2018, Jiaoxi, Yilan, Taiwan*, volume 123 of *LIPIcs*, pages 28:1–28:13. Schloss Dagstuhl - Leibniz-Zentrum für Informatik, 2018.

[14] P. Angelini, L. Cittadini, W. Didimo, F. Frati, G. Di Battista, M. Kaufmann, and A. Symvonis. On the perspectives opened by right angle crossing drawings. *J. Graph Algorithms Appl.*, 15(1):53–78, 2011.

[15] P. Angelini, D. Eppstein, F. Frati, M. Kaufmann, S. Lazard, T. Mchedlidze, M. Teillaud, and A. Wolff. Universal point sets for drawing planar graphs with circular arcs. *J. Graph Algorithms Appl.*, 18(3):313–324, 2014.

[16] P. Angelini, H. Förster, M. Hoffmann, M. Kaufmann, S. G. Kobourov, G. Liotta, and M. Patrignani. The QuaSEFE problem. In D. Archambault and C. D. Tóth, editors, *Graph Drawing and Network Visualization - 27th International Symposium, GD 2019, Prague, Czech Republic, September 17-20, 2019, Proceedings*, volume 11904 of *Lecture Notes in Computer Science*, pages 268–275. Springer, 2019.

[17] E. N. Argyriou, M. A. Bekos, and A. Symvonis. The straight-line RAC drawing problem is NP-hard. *J. Graph Algorithms Appl.*, 16(2):569–597, 2012.

[18] E. N. Argyriou, S. Cornelsen, H. Förster, M. Kaufmann, M. Nöllenburg, Y. Okamoto, C. N. Raftopoulou, and A. Wolff. Orthogonal and smooth orthogonal layouts of 1-planar graphs with low edge complexity. In T. C. Biedl and A. Kerren, editors, *Graph Drawing and Network Visualization -*

26th International Symposium, GD 2018, Barcelona, Spain, September 26-28, 2018, Proceedings, volume 11282 of *Lecture Notes in Computer Science*, pages 509–523. Springer, 2018.

[19] K. Arikushi, R. Fulek, B. Keszegh, F. Moric, and C. D. Tóth. Graphs that admit right angle crossing drawings. *Comput. Geom.*, 45(4):169–177, 2012.

[20] C. Auer, C. Bachmaier, F. J. Brandenburg, A. Gleißner, K. Hanauer, D. Neuwirth, and J. Reislhuber. Outer 1-planar graphs. *Algorithmica*, 74(4):1293–1320, 2016.

[21] C. Bachmaier, F. J. Brandenburg, K. Hanauer, D. Neuwirth, and J. Reislhuber. NIC-planar graphs. *Discret. Appl. Math.*, 232:23–40, 2017.

[22] S. W. Bae, J. Baffier, J. Chun, P. Eades, K. Eickmeyer, L. Grilli, S. Hong, M. Korman, F. Montecchiani, I. Rutter, and C. D. Tóth. Gap-planar graphs. *Theor. Comput. Sci.*, 745:36–52, 2018.

[23] M. J. Bannister, W. E. Devanny, V. Dujmović, D. Eppstein, and D. R. Wood. Track layouts, layered path decompositions, and leveled planarity. *Algorithmica*, 81(4):1561–1583, 2019.

[24] M. A. Bekos, W. Didimo, G. Liotta, S. Mehrabi, and F. Montecchiani. On RAC drawings of 1-planar graphs. *Theor. Comput. Sci.*, 689:48–57, 2017.

[25] M. A. Bekos, H. Förster, M. Gronemann, T. Mchedlidze, F. Montecchiani, C. N. Raftopoulou, and T. Ueckerdt. Planar graphs of bounded degree have bounded queue number. *SIAM J. Comput.*, 48(5):1487–1502, 2019.

[26] M. A. Bekos, H. Förster, and M. Kaufmann. On smooth orthogonal and octilinear drawings: Relations, complexity and Kandinsky drawings. *Algorithmica*, 81(5):2046–2071, 2019.

[27] M. A. Bekos, H. Förster, C. Geckeler, L. Holländer, M. Kaufmann, A. M. Spallek, and J. Splett. A Heuristic Approach Towards Drawings of Graphs With High Crossing Resolution. *The Computer Journal*, 11 2019.

[28] M. A. Bekos, M. Gronemann, M. Kaufmann, and R. Krug. Planar octilinear drawings with one bend per edge. *J. Graph Algorithms Appl.*, 19(2):657–680, 2015.

[29] M. A. Bekos, M. Gronemann, S. Pupyrev, and C. N. Raftopoulou. Perfect smooth orthogonal drawings. In N. G. Bourbakis, G. A. Tsihrintzis, and M. Virvou, editors, *5th International Conference on Information, Intelligence, Systems and Applications, IISA 2014, Chania, Crete, Greece, July 7-9, 2014*, pages 76–81. IEEE, 2014.

[30] M. A. Bekos, M. Kaufmann, F. Klute, S. Pupyrev, C. N. Raftopoulou, and T. Ueckerdt. Four pages are indeed necessary for planar graphs. *CoRR*, abs/2004.07630, 2020.

[31] M. A. Bekos, M. Kaufmann, S. G. Kobourov, and A. Symvonis. Smooth orthogonal layouts. *J. Graph Algorithms Appl.*, 17(5):575–595, 2013.

[32] M. A. Bekos, M. Kaufmann, and R. Krug. Sloggy drawings of graphs. In N. G. Bourbakis, G. A. Tsihrintzis, and M. Virvou, editors, *5th International Conference on Information, Intelligence, Systems and Applications, IISA 2014, Chania, Crete, Greece, July 7-9, 2014*, pages 82–87. IEEE, 2014.

[33] M. A. Bekos, M. Kaufmann, and R. Krug. Sloginsky drawings of graphs. In N. G. Bourbakis, G. A. Tsihrintzis, and M. Virvou, editors, *6th International Conference on Information, Intelligence, Systems and Applications, IISA 2015, Corfu, Greece, July 6-8, 2015*, pages 1–6. IEEE, 2015.

[34] M. A. Bekos, M. Kaufmann, and R. Krug. On the total number of bends for planar octilinear drawings. *J. Graph Algorithms Appl.*, 21(4):709–730, 2017.

[35] M. A. Bekos, M. Kaufmann, R. Krug, T. Ludwig, S. Näher, and V. Roselli. Slanted orthogonal drawings: Model, algorithms and evaluations. *J. Graph Algorithms Appl.*, 18(3):459–489, 2014.

[36] M. A. Bekos, M. Kaufmann, R. Krug, and M. Siebenhaller. The effect of almost-empty faces on planar Kandinsky drawings. In E. Bampis, editor, *Experimental Algorithms - 14th International Symposium, SEA 2015, Paris, France, June 29 - July 1, 2015, Proceedings*, volume 9125 of *Lecture Notes in Computer Science*, pages 352–364. Springer, 2015.

[37] M. A. Bekos, M. Kaufmann, and C. N. Raftopoulou. On the density of non-simple 3-planar graphs. In Y. Hu and M. Nöllenburg, editors, *Graph Drawing and Network Visualization - 24th International Symposium, GD 2016, Athens, Greece, September 19-21, 2016, Revised Selected Papers*, volume 9801 of *Lecture Notes in Computer Science*, pages 344–356. Springer, 2016.

[38] M. A. Bekos, M. Kaufmann, and C. N. Raftopoulou. On optimal 2- and 3-planar graphs. In B. Aronov and M. J. Katz, editors, *33rd International Symposium on Computational Geometry, SoCG 2017, July 4-7, 2017, Brisbane, Australia*, volume 77 of *LIPIcs*, pages 16:1–16:16. Schloss Dagstuhl - Leibniz-Zentrum für Informatik, 2017.

[39] M. A. Bekos, M. Kaufmann, and C. Zielke. The book embedding problem from a SAT-solving perspective. In E. D. Giacomo and A. Lubiw, editors, *Graph Drawing and Network Visualization - 23rd International Symposium,*

GD 2015, Los Angeles, CA, USA, September 24-26, 2015, Revised Selected Papers, volume 9411 of *Lecture Notes in Computer Science*, pages 125–138. Springer, 2015.

[40] F. Bernhart and P. C. Kainen. The book thickness of a graph. *J. Comb. Theory, Ser. B*, 27(3):320–331, 1979.

[41] P. Bertolazzi, G. Di Battista, and W. Didimo. Computing orthogonal drawings with the minimum number of bends. *IEEE Trans. Computers*, 49(8):826–840, 2000.

[42] S. N. Bhatt, F. R. K. Chung, F. T. Leighton, and A. L. Rosenberg. Scheduling tree-dags using FIFO queues: A control-memory trade-off. *J. Parallel Distributed Comput.*, 33(1):55–68, 1996.

[43] T. C. Biedl and G. Kant. A better heuristic for orthogonal graph drawings. *Comput. Geom.*, 9(3):159–180, 1998.

[44] T. Bläsius, M. Krug, I. Rutter, and D. Wagner. Orthogonal graph drawing with flexibility constraints. *Algorithmica*, 68(4):859–885, 2014.

[45] T. Bläsius, S. Lehmann, and I. Rutter. Orthogonal graph drawing with inflexible edges. *Comput. Geom.*, 55:26–40, 2016.

[46] O. V. Borodin. On acyclic colorings of planar graphs. *Discret. Math.*, 306(10-11):953–972, 2006.

[47] F. J. Brandenburg. 1-visibility representations of 1-planar graphs. *J. Graph Algorithms Appl.*, 18(3):421–438, 2014.

[48] F. J. Brandenburg, W. Didimo, W. S. Evans, P. Kindermann, G. Liotta, and F. Montecchiani. Recognizing and drawing IC-planar graphs. *Theor. Comput. Sci.*, 636:1–16, 2016.

[49] G. Brinkmann and B. D. McKay. Fast generation of some classes of planar graphs. *Electron. Notes Discret. Math.*, 3:28–31, 1999.

[50] C. Buchheim, M. Chimani, C. Gutwenger, M. Jünger, and P. Mutzel. Crossings and planarization. In R. Tamassia, editor, *Handbook on Graph Drawing and Visualization*, pages 43–85. Chapman and Hall/CRC, 2013.

[51] J. Cardinal, M. Hoffmann, V. Kusters, C. D. Tóth, and M. Wettstein. Arc diagrams, flip distances, and Hamiltonian triangulations. *Comput. Geom.*, 68:206–225, 2018.

[52] S. Chaplick, H. Förster, M. Hoffmann, and M. Kaufmann. Monotone arc diagrams with few biarcs. *CoRR*, abs/2003.05332, 2020. An extended abstract has been presented at EuroCG 2020.

[53] S. Chaplick, H. Förster, M. Kryven, and A. Wolff. On arrangements of orthogonal circles. In D. Archambault and C. D. Tóth, editors, *Graph Drawing and Network Visualization - 27th International Symposium, GD 2019, Prague, Czech Republic, September 17-20, 2019, Proceedings*, volume 11904 of *Lecture Notes in Computer Science*, pages 216–229. Springer, 2019.

[54] S. Chaplick, H. Förster, M. Kryven, and A. Wolff. Drawing graphs with circular arcs and right-angle crossings. In S. Albers, editor, *17th Scandinavian Symposium and Workshops on Algorithm Theory, SWAT 2020, June 22-24, 2020, Tórshavn, Faroe Islands*, volume 162 of *LIPIcs*, pages 21:1–21:14. Schloss Dagstuhl - Leibniz-Zentrum für Informatik, 2020.

[55] S. Chaplick, F. Lipp, A. Wolff, and J. Zink. Compact drawings of 1-planar graphs with right-angle crossings and few bends. *Comput. Geom.*, 84:50–68, 2019.

[56] O. Cheong, S. Har-Peled, H. Kim, and H. Kim. On the number of edges of fan-crossing free graphs. *Algorithmica*, 73(4):673–695, 2015.

[57] F. R. K. Chung, F. T. Leighton, and A. L. Rosenberg. Diogenes: a methodology for designing fault-tolerant VLSI processor arrays. In *Proceedings of the Thirteenth Fault-Tolerant Computer Symposium*, 1983.

[58] F. R. K. Chung, F. T. Leighton, and A. L. Rosenberg. Embedding graphs in books: A layout problem with applications to VLSI design. *SIAM Journal on Algebraic Discrete Methods*, 8(1):33–58, 1987.

[59] T. H. Cormen, C. E. Leiserson, R. L. Rivest, and C. Stein. *Introduction to Algorithms, Second Edition*. The MIT Press and McGraw-Hill Book Company, 2001.

[60] H. de Fraysseix, J. Pach, and R. Pollack. How to draw a planar graph on a grid. *Combinatorica*, 10(1):41–51, 1990.

[61] H. R. Dehkordi and P. Eades. Every outer-1-plane graph has a right angle crossing drawing. *Int. J. Comput. Geometry Appl.*, 22(6):543–558, 2012.

[62] G. Di Battista, W. Didimo, M. Patrignani, and M. Pizzonia. Orthogonal and quasi-upward drawings with vertices of prescribed size. In J. Kratochvíl, editor, *Graph Drawing, 7th International Symposium, GD'99, Stirín Castle, Czech Republic, September 1999, Proceedings*, volume 1731 of *Lecture Notes in Computer Science*, pages 297–310. Springer, 1999.

[63] G. Di Battista, P. Eades, R. Tamassia, and I. G. Tollis. *Graph Drawing: Algorithms for the Visualization of Graphs*. Prentice-Hall, 1999.

[64] E. Di Giacomo, W. Didimo, P. Eades, and G. Liotta. 2-layer right angle crossing drawings. *Algorithmica*, 68(4):954–997, 2014.

[65] E. Di Giacomo, W. Didimo, and G. Liotta. Spine and radial drawings. In R. Tamassia, editor, *Handbook on Graph Drawing and Visualization*, pages 247–284. Chapman and Hall/CRC, 2013.

[66] E. Di Giacomo, W. Didimo, G. Liotta, and H. Meijer. Area, curve complexity, and crossing resolution of non-planar graph drawings. *Theory Comput. Syst.*, 49(3):565–575, 2011.

[67] E. Di Giacomo, W. Didimo, G. Liotta, and S. K. Wismath. Curve-constrained drawings of planar graphs. *Comput. Geom.*, 30(1):1–23, 2005.

[68] E. Di Giacomo, G. Liotta, and H. Meijer. Computing straight-line 3D grid drawings of graphs in linear volume. *Comput. Geom.*, 32(1):26–58, 2005.

[69] E. Di Giacomo, G. Liotta, and F. Montecchiani. The planar slope number of subcubic graphs. In A. Pardo and A. Viola, editors, *LATIN 2014: Theoretical Informatics - 11th Latin American Symposium, Montevideo, Uruguay, March 31 - April 4, 2014. Proceedings*, volume 8392 of *Lecture Notes in Computer Science*, pages 132–143. Springer, 2014.

[70] W. Didimo, P. Eades, and G. Liotta. A characterization of complete bipartite RAC graphs. *Inf. Process. Lett.*, 110(16):687–691, 2010.

[71] W. Didimo, P. Eades, and G. Liotta. Drawing graphs with right angle crossings. *Theor. Comput. Sci.*, 412(39):5156–5166, 2011.

[72] W. Didimo, G. Liotta, and F. Montecchiani. A survey on graph drawing beyond planarity. *ACM Comput. Surv.*, 52(1):4:1–4:37, 2019.

[73] W. Didimo, G. Liotta, and M. Patrignani. On the complexity of HV-rectilinear planarity testing. In C. A. Duncan and A. Symvonis, editors, *Graph Drawing - 22nd International Symposium, GD 2014, Würzburg, Germany, September 24-26, 2014, Revised Selected Papers*, volume 8871 of *Lecture Notes in Computer Science*, pages 343–354. Springer, 2014.

[74] R. Diestel. *Graph Theory, 5th Edition*, volume 173 of *Graduate texts in mathematics*. Springer, 2017.

[75] V. Dujmović and F. Frati. Stack and queue layouts via layered separators. *J. Graph Algorithms Appl.*, 22(1):89–99, 2018.

[76] V. Dujmović, G. Joret, P. Micek, P. Morin, T. Ueckerdt, and D. R. Wood. Planar graphs have bounded queue-number. In D. Zuckerman, editor, *60th IEEE Annual Symposium on Foundations of Computer Science, FOCS 2019,*

Baltimore, Maryland, USA, November 9-12, 2019, pages 862–875. IEEE Computer Society, 2019.

[77] V. Dujmović, P. Morin, and D. R. Wood. Path-width and three-dimensional straight-line grid drawings of graphs. In S. G. Kobourov and M. T. Goodrich, editors, *Graph Drawing, 10th International Symposium, GD 2002, Irvine, CA, USA, August 26-28, 2002, Revised Papers*, volume 2528 of *Lecture Notes in Computer Science*, pages 42–53. Springer, 2002.

[78] V. Dujmović, P. Morin, and D. R. Wood. Layout of graphs with bounded tree-width. *SIAM J. Comput.*, 34(3):553–579, 2005.

[79] V. Dujmović, A. Pór, and D. R. Wood. Track layouts of graphs. *Discret. Math. Theor. Comput. Sci.*, 6(2):497–522, 2004.

[80] V. Dujmović and D. R. Wood. Three-dimensional grid drawings with sub-quadratic volume. In G. Liotta, editor, *Graph Drawing, 11th International Symposium, GD 2003, Perugia, Italy, September 21-24, 2003, Revised Papers*, volume 2912 of *Lecture Notes in Computer Science*, pages 190–201. Springer, 2003.

[81] V. Dujmović and D. R. Wood. Stacks, queues and tracks: Layouts of graph subdivisions. *Discret. Math. Theor. Comput. Sci.*, 7(1):155–202, 2005.

[82] C. A. Duncan, D. Eppstein, M. T. Goodrich, S. G. Kobourov, and M. Nöllenburg. Lombardi drawings of graphs. *J. Graph Algorithms Appl.*, 16(1):85–108, 2012.

[83] S. Durocher, S. Felsner, S. Mehrabi, and D. Mondal. Drawing HV-restricted planar graphs. In A. Pardo and A. Viola, editors, *LATIN 2014: Theoretical Informatics - 11th Latin American Symposium, Montevideo, Uruguay, March 31 - April 4, 2014. Proceedings*, volume 8392 of *Lecture Notes in Computer Science*, pages 156–167. Springer, 2014.

[84] P. Eades and G. Liotta. Right angle crossing graphs and 1-planarity. *Discret. Appl. Math.*, 161(7-8):961–969, 2013.

[85] R. B. Eggleton. Rectilinear drawings of graphs. *Utilitas Math.*, 29:149–172, 1986.

[86] D. Eppstein. Planar Lombardi drawings for subcubic graphs. In W. Didimo and M. Patrignani, editors, *Graph Drawing - 20th International Symposium, GD 2012, Redmond, WA, USA, September 19-21, 2012, Revised Selected Papers*, volume 7704 of *Lecture Notes in Computer Science*, pages 126–137. Springer, 2012.

[87] L. Euler. Demonstratio nonnullarum insignium proprietatum, quibus solida hedris planis inclusa sunt praedita. *Novi Commentarii academiae scientiarum Petropolitanae*, 4:140–160, 1758.

[88] L. Euler. Elementa doctrinae solidorum. *Novi Commentarii academiae scientiarum Petropolitanae*, 4:109–140, 1758.

[89] H. Everett, S. Lazard, G. Liotta, and S. K. Wismath. Universal sets of n points for one-bend drawings of planar graphs with n vertices. *Discret. Comput. Geom.*, 43(2):272–288, 2010.

[90] S. Felsner. *Geometric Graphs and Arrangements - Some Chapters from Combinatorial Geometry*. Advanced lectures in mathematics. Vieweg+Teubner, 2004.

[91] S. Felsner, G. Liotta, and S. K. Wismath. Straight-line drawings on restricted integer grids in two and three dimensions. *J. Graph Algorithms Appl.*, 7(4):363–398, 2003.

[92] H. J. Fleischner, D. P. Geller, and F. Harary. Outerplanar graphs and weak duals. *The Journal of the Indian Mathematical Society*, 38(1-4):215–219, 1974.

[93] H. Förster, R. Ganian, F. Klute, and M. Nöllenburg. On strict (outer-)confluent graphs. In D. Archambault and C. D. Tóth, editors, *Graph Drawing and Network Visualization - 27th International Symposium, GD 2019, Prague, Czech Republic, September 17-20, 2019, Proceedings*, volume 11904 of *Lecture Notes in Computer Science*, pages 147–161. Springer, 2019.

[94] H. Förster and M. Kaufmann. On compact RAC drawings. In *ESA 2020*, volume 173. LIPIcs, 2020. To be published. The results also have been announced in a poster at GD2019.

[95] U. Fößmeier and M. Kaufmann. Drawing high degree graphs with low bend numbers. In F. Brandenburg, editor, *Graph Drawing, Symposium on Graph Drawing, GD '95, Passau, Germany, September 20-22, 1995, Proceedings*, volume 1027 of *Lecture Notes in Computer Science*, pages 254–266. Springer, 1995.

[96] U. Fößmeier and M. Kaufmann. Algorithms and area bounds for nonplanar orthogonal drawings. In G. D. Battista, editor, *Graph Drawing, 5th International Symposium, GD '97, Rome, Italy, September 18-20, 1997, Proceedings*, volume 1353 of *Lecture Notes in Computer Science*, pages 134–145. Springer, 1997.

[97] J. Fox, J. Pach, and A. Suk. The number of edges in k-quasi-planar graphs. *SIAM J. Discrete Math.*, 27(1):550–561, 2013.

[98] J. L. Ganley. Stack and queue layouts of Halin graphs, 1995. `http://www.ganley.org/pubs/Halin.pdf`. Unpublished.

[99] M. R. Garey and D. S. Johnson. *Computers and Intractability: A Guide to the Theory of NP-Completeness*. W. H. Freeman, 1979.

[100] A. Garg and R. Tamassia. On the computational complexity of upward and rectilinear planarity testing. *SIAM J. Comput.*, 31(2):601–625, 2001.

[101] A. Grigoriev and H. L. Bodlaender. Algorithms for graphs embeddable with few crossings per edge. *Algorithmica*, 49(1):1–11, 2007.

[102] M. Gronemann. Bitonic st-orderings of biconnected planar graphs. In C. A. Duncan and A. Symvonis, editors, *Graph Drawing - 22nd International Symposium, GD 2014, Würzburg, Germany, September 24-26, 2014, Revised Selected Papers*, volume 8871 of *Lecture Notes in Computer Science*, pages 162–173. Springer, 2014.

[103] B. Grünbaum. *Convex Polytopes*. Interscience, Wiley, London, 1967.

[104] C. Gutwenger and P. Mutzel. Planar polyline drawings with good angular resolution. In S. Whitesides, editor, *Graph Drawing, 6th International Symposium, GD'98, Montréal, Canada, August 1998, Proceedings*, volume 1547 of *Lecture Notes in Computer Science*, pages 167–182. Springer, 1998.

[105] F. Harary and A. J. Schwenk. The number of caterpillars. *Discret. Math.*, 6(4):359–365, 1973.

[106] D. Harel and M. Sardas. An algorithm for straight-line drawing of planar graphs. *Algorithmica*, 20(2):119–135, 1998.

[107] L. S. Heath, F. T. Leighton, and A. L. Rosenberg. Comparing queues and stacks as mechanisms for laying out graphs. *SIAM J. Discrete Math.*, 5(3):398–412, 1992.

[108] L. S. Heath and A. L. Rosenberg. Laying out graphs using queues. *SIAM J. Comput.*, 21(5):927–958, 1992.

[109] S. Hong, P. Eades, N. Katoh, G. Liotta, P. Schweitzer, and Y. Suzuki. A linear-time algorithm for testing outer-1-planarity. *Algorithmica*, 72(4):1033–1054, 2015.

[110] S. Hong, D. Merrick, and H. A. D. do Nascimento. Automatic visualisation of metro maps. *J. Vis. Lang. Comput.*, 17(3):203–224, 2006.

[111] S. Hong and H. Nagamochi. A linear-time algorithm for testing full outer-2-planarity. *Discret. Appl. Math.*, 255:234–257, 2019.

[112] S.-H. Hong, P. Eades, G. Liotta, and S.-H. Poon. Fáry's theorem for 1-planar graphs. In J. Gudmundsson, J. Mestre, and T. Viglas, editors, *Computing and Combinatorics*, pages 335–346, Berlin, Heidelberg, 2012. Springer Berlin Heidelberg.

[113] W. Huang. Using eye tracking to investigate graph layout effects. In S. Hong and K. Ma, editors, *APVIS 2007, 6th International Asia-Pacific Symposium on Visualization 2007, Sydney, Australia, 5-7 February 2007*, pages 97–100. IEEE Computer Society, 2007.

[114] W. Huang, P. Eades, and S. Hong. A graph reading behavior: Geodesic-path tendency. In P. Eades, T. Ertl, and H. Shen, editors, *IEEE Pacific Visualization Symposium PacificVis 2009, Beijing, China, April 20-23, 2009*, pages 137–144. IEEE Computer Society, 2009.

[115] W. Huang, P. Eades, and S. Hong. Larger crossing angles make graphs easier to read. *J. Vis. Lang. Comput.*, 25(4):452–465, 2014.

[116] G. Kant. Hexagonal grid drawings. In E. W. Mayr, editor, *Graph-Theoretic Concepts in Computer Science, 18th International Workshop, WG '92, Wiesbaden-Naurod, Germany, June 19-20, 1992, Proceedings*, volume 657 of *Lecture Notes in Computer Science*, pages 263–276. Springer, 1992.

[117] G. Kant. Drawing planar graphs using the canonical ordering. *Algorithmica*, 16(1):4–32, 1996.

[118] M. Kaufmann and T. Ueckerdt. The density of fan-planar graphs. *CoRR*, abs/1403.6184, 2014.

[119] M. Kaufmann and D. Wagner, editors. *Drawing Graphs, Methods and Models (the book grow out of a Dagstuhl Seminar, April 1999)*, volume 2025 of *Lecture Notes in Computer Science*. Springer, 2001.

[120] M. Kaufmann and R. Wiese. Embedding vertices at points: Few bends suffice for planar graphs. *J. Graph Algorithms Appl.*, 6(1):115–129, 2002.

[121] B. Keszegh, J. Pach, and D. Pálvölgyi. Drawing planar graphs of bounded degree with few slopes. *SIAM J. Discrete Math.*, 27(2):1171–1183, 2013.

[122] P. Kindermann, F. Montecchiani, L. Schlipf, and A. Schulz. Drawing subcubic 1-planar graphs with few bends, few slopes, and large angles. In T. C. Biedl and A. Kerren, editors, *Graph Drawing and Network Visualization - 26th International Symposium, GD 2018, Barcelona, Spain, September 26-28, 2018, Proceedings*, volume 11282 of *Lecture Notes in Computer Science*, pages 152–166. Springer, 2018.

[123] S. G. Kobourov, G. Liotta, and F. Montecchiani. An annotated bibliography on 1-planarity. *Computer Science Review*, 25:49–67, 2017.

[124] F. T. Leighton. *Complexity Issues in VLSI: Optimal Layouts for the Shuffle-Exchange Graph and Other Networks*. MIT Press, Cambridge, MA, USA, 1983.

[125] F. T. Leighton. New lower bound techniques for VLSI. *Mathematical Systems Theory*, 17(1):47–70, 1984.

[126] F. T. Leighton and A. L. Rosenberg. Three-dimensional circuit layouts. *SIAM J. Comput.*, 15(3):793–813, 1986.

[127] C. E. Leiserson. Area-efficient graph layouts (for VLSI). In *21st Annual Symposium on Foundations of Computer Science, Syracuse, New York, USA, 13-15 October 1980*, pages 270–281. IEEE Computer Society, 1980.

[128] A. Lempel, S. Even, and I. Cederbaum. An algorithm for planarity testing of graphs. In *Theory of Graphs (Internat. Sympos., Rome, 1966)*, pages 215–232. Gordon and Breach, New York; Dunod, Paris, 1967.

[129] Y. Liu, A. Morgana, and B. Simeone. A linear algorithm for 2-bend embeddings of planar graphs in the two-dimensional grid. *Discrete Applied Mathematics*, 81(1-3):69–91, 1998.

[130] M. Löffler and C. D. Tóth. Linear-size universal point sets for one-bend drawings. In E. D. Giacomo and A. Lubiw, editors, *Graph Drawing and Network Visualization - 23rd International Symposium, GD 2015, Los Angeles, CA, USA, September 24-26, 2015, Revised Selected Papers*, volume 9411 of *Lecture Notes in Computer Science*, pages 423–429. Springer, 2015.

[131] J. Manuch, M. Patterson, S. Poon, and C. Thachuk. Complexity of finding non-planar rectilinear drawings of graphs. In U. Brandes and S. Cornelsen, editors, *Graph Drawing - 18th International Symposium, GD 2010, Konstanz, Germany, September 21-24, 2010. Revised Selected Papers*, volume 6502 of *Lecture Notes in Computer Science*, pages 305–316. Springer, 2010.

[132] T. A. J. Nicholson. Permutation procedure for minimising the number of crossings in a network. *Proceedings of the Institution of Electrical Engineers*, 115(1):21–26, January 1968.

[133] T. Nishizeki and M. S. Rahman. *Planar Graph Drawing*, volume 12 of *Lecture Notes Series on Computing*. World Scientific, 2004.

[134] M. Nöllenburg and A. Wolff. Drawing and labeling high-quality metro maps by mixed-integer programming. *IEEE Trans. Vis. Comput. Graph.*, 17(5):626–641, 2011.

[135] M. Nöllenburg. Automated drawing of metro maps. Master's thesis, Fakultät für Informatik, Universität Karlsruhe (TH), Aug. 2005.

[136] J. Pach, R. Radoicic, G. Tardos, and G. Tóth. Improving the crossing lemma by finding more crossings in sparse graphs. *Discret. Comput. Geom.*, 36(4):527–552, 2006.

[137] J. Pach and G. Tóth. Graphs drawn with few crossings per edge. *Combinatorica*, 17(3):427–439, 1997.

[138] M. Patrignani. Planarity testing and embedding. In R. Tamassia, editor, *Handbook on Graph Drawing and Visualization*, pages 1–42. Chapman and Hall/CRC, 2013.

[139] S. Pupyrev. Mixed linear layouts of planar graphs. In F. Frati and K. Ma, editors, *Graph Drawing and Network Visualization - 25th International Symposium, GD 2017, Boston, MA, USA, September 25-27, 2017, Revised Selected Papers*, volume 10692 of *Lecture Notes in Computer Science*, pages 197–209. Springer, 2017.

[140] H. C. Purchase. Effective information visualisation: a study of graph drawing aesthetics and algorithms. *Interacting with Computers*, 13(2):147–162, 2000.

[141] H. C. Purchase, D. A. Carrington, and J. Allder. Empirical evaluation of aesthetics-based graph layout. *Empirical Software Engineering*, 7(3):233–255, 2002.

[142] Z. Rahmati and F. Emami. RAC drawings in subcubic area. *Information Processing Letters*, 159-160:105945, 2020.

[143] S. Rengarajan and C. E. V. Madhavan. Stack and queue number of 2-trees. In D. Du and M. Li, editors, *Computing and Combinatorics, First Annual International Conference, COCOON '95, Xi'an, China, August 24-26, 1995, Proceedings*, volume 959 of *Lecture Notes in Computer Science*, pages 203–212. Springer, 1995.

[144] G. Ringel. Ein Sechsfarbenproblem auf der Kugel. *Abhandlungen aus dem Mathematischen Seminar der Universität Hamburg*, 29(1):107–117, Dec 1965.

[145] P. Rosenstiehl and R. E. Tarjan. Rectilinear planar layouts and bipolar orientations of planar graphs. *Discrete & Computational Geometry*, 1:343–353, 1986.

[146] T. L. Saaty. The minimum number of intersections in complete graphs. *Proceedings of the National Academy of Sciences of the United States of America*, 52(3):688–690, 1964.

[147] W. Schnyder. Embedding planar graphs on the grid. In D. S. Johnson, editor, *Proceedings of the First Annual ACM-SIAM Symposium on Discrete Algorithms, 22-24 January 1990, San Francisco, California, USA*, pages 138–148. SIAM, 1990.

[148] J. M. Stott, P. Rodgers, J. C. Martinez-Ovando, and S. G. Walker. Automatic metro map layout using multicriteria optimization. *IEEE Trans. Vis. Comput. Graph.*, 17(1):101–114, 2011.

[149] R. Tamassia. On embedding a graph in the grid with the minimum number of bends. *SIAM J. Comput.*, 16(3):421–444, 1987.

[150] R. Tamassia, editor. *Handbook on Graph Drawing and Visualization*. Chapman and Hall/CRC, 2013.

[151] R. Tamassia and I. G. Tollis. A unified approach a visibility representation of planar graphs. *Discrete & Computational Geometry*, 1:321–341, 1986.

[152] R. Tamassia and I. G. Tollis. Planar grid embedding in linear time. *IEEE Transactions on Circuits and Systems*, 36(9):1230–1234, Sep. 1989.

[153] R. E. Tarjan. Sorting using networks of queues and stacks. *J. ACM*, 19(2):341–346, 1972.

[154] C. Thomassen. Rectilinear drawings of graphs. *Journal of Graph Theory*, 12(3):335–341, 1988.

[155] J. C. Urschel and J. Wellens. Testing k-planarity is NP-complete. *CoRR*, abs/1907.02104, 2019.

[156] L. G. Valiant. Universality considerations in VLSI circuits. *IEEE Trans. Computers*, 30(2):135–140, 1981.

[157] L. Vismara. Planar straight-line drawing algorithms. In R. Tamassia, editor, *Handbook on Graph Drawing and Visualization*, pages 193–222. Chapman and Hall/CRC, 2013.

[158] V. Wiechert. On the queue-number of graphs with bounded tree-width. *Electr. J. Comb.*, 24(1):P1.65, 2017.

[159] R. J. Wilson. *Introduction to Graph Theory*. Addison Wesley Longman Limited, Edinburgh Gate, Harlow, Essex CM20 2JE, England, 4th edition, 1996.

[160] D. R. Wood. Bounded-degree graphs have arbitrarily large queue-number. *Discret. Math. Theor. Comput. Sci.*, 10(1), 2008.

[161] M. Yannakakis. Embedding planar graphs in four pages. *J. Comput. Syst. Sci.*, 38(1):36–67, 1989.

Appendix A

Other Works of the Author

Here, we provide an annotated list of other collaborative works of the author that were published in peer-reviewed conferences during the thesis period:

– In [11], we investigate a special type of st-orderings called *bitonic*. In a bitonic st-ordering of a planar graph, the st-numbers of the successors of each vertex form a bitonic list in the ordering induced by some planar embedding. Bitonic st-orderings can for instance be used to create upward graph drawings. It is noteworthy that not every directed planar graph admits such an ordering. However, by subdividing each edge at most once, a graph with a bitonic st-ordering can be obtained. We show how to compute the set of edges of minimal cardinality whose subdivision results in a graph with a bitonic st-ordering over all embeddings.

– In [16], we investigate a variant of the simultaneous graph drawing problem called *QuaSEFE*. In simultaneous graph drawing, multiple graphs on a shared vertex set have to be drawn. In the QuaSEFE model, each such graph has to be drawn quasiplanar. The main difficulty here is that edges may appear in more than one of the graphs but must be realized in the same way in each of them. We identify several sufficient conditions for positive instances and present restrictive negative instances.

– In [27], we approach the *LAC drawing* problem from a practical point-of-view. Namely, we provide a probalistic hill-climbing method that given an initial drawing produces drawings with better crossing, angular or total resolution. Using a data set of several thousand graphs, we verify that this goal can indeed be achieved.

– In [53], we investigate *arrangements of orthogonal circles*, that is, circles in the arrangement can be either disjoint or intersect at right angles only. We prove that the number of faces in such arrangements is linear which stands in contrast to general arrangements of circles. This result also directly transfers to the

number of edges in an *orthogonal circle intersection graph* in which vertices are represented by circles and edges by orthogonal intersections between circles, Finally, we prove NP-hardness for the recognition of such graphs under the constraint that all circles have a unit size. In a follow-up study [54], we also investigate the class of *arc-RAC graphs* which admit RAC drawings where every edge is represented by a circular arc and provide a density upper and lower bound.

– In [93], we investigate *strict confluent drawings*. In a confluent drawing, edges are smooth sequences of segments between vertices and/or junctions where junctions may be regarded as Δ-junctions (that is, three segments are smoothly connected in a pairwise fashion) or merge/split junctions (that is, one segment is smoothly connected to two segments which are not connected by a smooth curve). In a strict drawing, each edge is represented by exactly one smooth curve. We consider relationships to several types of intersection graphs and discuss other properties of graphs admitting strict confluent drawings.

Index