Restricted String Representations

by

Martin Derka

A thesis
presented to the University of Waterloo
in fulfillment of the
thesis requirement for the degree of
Doctor of Philosophy
in
Computer Science

Waterloo, Ontario, Canada, 2017

Examining Committee Membership

The following served on the Examining Committee for this thesis. The decision of the Examining Committee is by majority vote.

External Examiner	David Eppstein Professor
Supervisor	Therese Biedl Professor
Internal Member	Anna Lubiw Professor
Internal Member	Lap Chi Lau Associate Professor
Internal-External Member	Bruce Richter Professor

I hereby declare that I am the sole author of this thesis. This is a true copy of the thesis, including any required final revisions, as accepted by my examiners.

I understand that my thesis may be made electronically available to the public.

Abstract

A string representation of a graph assigns to every vertex a curve in the plane so that two curves intersect if and only if the represented vertices are adjacent. This work investigates string representations of graphs with an emphasis on the shapes of curves and the way they intersect. We strengthen some previously known results and show that every planar graph has string representations where every curve consists of axis-parallel line segments with at most two bends (those are the so-called B_2-VPG representations) and simultaneously two curves intersect each other at most once (those are the so-called 1-string representations). Thus, planar graphs are B_2-VPG 1-string graphs. We further show that with some restrictions on the shapes of the curves, string representations can be used to produce approximation algorithms for several hard problems. The B_2-VPG representations of planar graphs satisfy these restrictions. We attempt to further restrict the number of bends in VPG representations for subclasses of planar graphs, and investigate B_1-VPG representations. We propose new classes of string representations for planar graphs that we call "order-preserving." Order-preservation is an interesting property which relates the string representation to the planar embedding of the graph, and we believe that it might prove useful when constructing string representations. Finally, we extend our investigation of string representations to string representations that require some curves to intersect multiple times. We show that there are outer-string graphs that require an exponential number of crossings in their outer-string representations. Our construction also proves that 1-planar graphs, i.e., graphs that are no longer planar, yet fairly close to planar graphs, may have string representations, but they are not always 1-string.

Acknowledgements

First and foremost, I would like to thank my supervisor, Prof. Therese Biedl, for giving me the opportunity to pursue my Ph.D. degree at the University of Waterloo. Therese is one of the best researchers that I ever met, and I am grateful for the opportunity to learn from her. I highly appreciate and value her guidance, expertise, attention to detail and all the time that she invested in me. I would also like to thank my thesis committee, professors Anna Lubiw, Bruce Richter, Lap Chi Lau and David Eppstein for their valuable remarks and discussion.

A special thank you belongs to Prof. Alejandro López-Ortiz. Alex was one of the first people I met at the University of Waterloo. He became my dear friend, collaborator, advisor, and mentor. I will forever highly regard him for all the influence he had on my life here in Waterloo. Alex passed away after a long and brave battle with an illness on Sunday, March 12, 2017, and I am very sad that he cannot see the result of my work that he helped to shape.

I would like to thank Wendy Rush who was always around to save me when I was lost. Wendy was the person who I would go ask for opening my office every time I locked myself out early in the morning. She would be the saviour who gets our printer working when it decides not, offers help with booking rooms, and spends endless hours helping with travel refund claims and paperwork that I do not know how to handle myself.

I would like to acknowledge and thank Prof. Petr Hliněný, my undergraduate and graduate supervisor from Brno, for seeing my potential back in the early days, raising my interest in graph theory and helping to form my career path. I also thank professors Markus Chimani, Drago Bokal, Gelasio Salazar, and Jan Kratochvíl for broadening my horizons with many fruitful discussions, both of academic and leisure nature. Last, but not least, I would like to say thank you to Prof. Ian Munro for being my friend and offering words of advice whenever needed.

I would further like to thank all the fellow students and my friends in the algorithms and complexity group for helping to create motivating and supporting environment for me and helping me to stay sane and during this long run. Those are especially Shahin Kamali,

Daniela Maftuleac, Saeed Mehrabi, Hisham El-Zein, Matthew Robertson, Simon Pratt, Oliver Grant, Camila Pérez Gavilán and Amit Levi. I would also like to thank the friends from my soccer team, and especially our dedicated captain, organizer, and great player, Rafael Olaechea, for the numerous evenings of fun that we enjoyed together.

The biggest thank you of all belongs to my girlfriend Stephanie Zahorka for being the love of my life. Stephanie patiently helped me to get through the long streaks of work while chasing deadlines, brewed tons of cups of coffee during the long nights of writing papers, and morally supported me during my entire studies. My thanks also belongs to my parents and sister for their love and encouragement throughout my life.

Finally, I would like to thank NSERC and Vanier CGS for supporting my studies.

Table of Contents

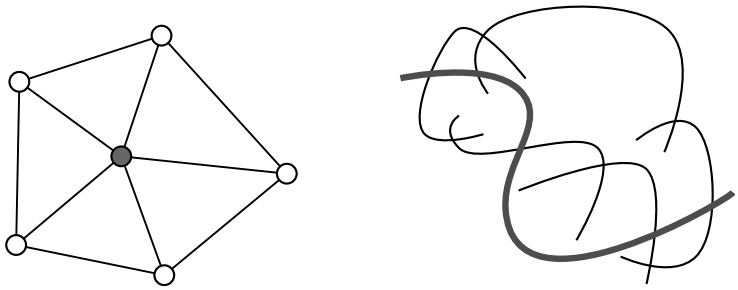
Chapter 1

Introduction

Graphs are an abstract way of describing networks, maps, diagrams, geometric objects, or anything else that consists of sites (vertices, nodes, points, ...) and connections (edges, arcs, ...). The usual visual representation of a graph is drawing the sites as points in the plane and drawing a connection between two sitest as a curve between the corresponding points. A *string representation* of a graph is a different way of representing a graph when the sites themselves become curves drawn in the plane, and there is a connection between two sites if and only if their curves intersect. Graphs that have string representations are called *string graphs*. See Figure 1.1.

String graphs have many applications in civil, electrical and computer engineering. For instance, in VLSI design, we need to transfer signals via channels embedded into circuit boards. The channels on their own cannot cross and therefore, they are embedded into circuit boards in several crossing-free layers. There is a strong desire to reduce the number of layers as it reduces the manufacturing cost. Minimizing the number of layers is the same thing as solving the colouring problem on a string graph. In consequence, one wonders what graphs can be string graphs. As a further restriction, the shapes of the conductive channels cannot be arbitrary, since only some angles are allowed when the channels need to be bent. This motivates the main question of the thesis: What graphs have a string representation that satisfies certain restrictions on the strings?

For another example of an application, imagine a manufacturing plant which is partly



(a) A graph. (b) A string representation.

Figure 1.1: A graph with a string representation.

operated by autonomous robots. The robots are mobile and have set routes along which they can move. The network of all such routes forms a string graph. Selecting the maximum set of robots that can move along the routes at the same time without any possibility of colliding is the same as selecting the maximum independent set in such a string graph. The problem is NP-hard in general graphs, and even NP-hard to approximate within a factor of $o(n^{\epsilon})$, but as we show in this work, a suitable set of restrictions on the shapes of the curves in a string representation makes the problem approximable in polynomial time within a factor of $O(\log n)$.

Finally, string graphs are interesting from the theoretical perspective as they have strong relations to crossing numbers and rotation systems of graphs. One of the immediately related questions is whether a given rotation system of a complete graph K_n has a realization in which no pair of incident edges intersects (this is so-called semi-simple drawing [4]). This can be rephrased as a problem of realizing a string graph.

Main questions. It is well known that there are graphs that do not have string representations, i.e., not every graph is a string graph. The only known class of graphs without string representations is the class of graphs that contain a full subdivision of a non-planar graph as an induced minor. Yet, there is no proof that all the other graphs are string graphs. Thus, one of the main questions in this field is what kind of graph classes always

have (or never have) string representations. We investigate string representations when some natural restrictions (such as those stemming from the VLSI design) on the shape of curves and the way they intersect are imposed. We investigate their existence, relationships with other graph classes, and also algorithmic improvements that such representations yield. A detailed overview of the thesis organization is provided in Section 2.3 once the main definitions have been provided.

Chapter 2

Definitions and Preliminaries

2.1 Graph-theoretic preliminaries

A *graph* G is a pair (V, E) where V is a set of *vertices* and E is a set of vertex pairs called *edges*. The number of vertices is commonly denoted by n and the number of edges by m. If $e = \{u, v\}$ is an edge of G, we sometimes write $e = uv$ and say that u and v are the *ends* of e. We also say that e is *incident* to u and v, that u is a *neighbour* of v, and that u and v are *adjacent*. The *neighbourhood* of v is the set of all neighbours of v and it is denoted by $N(v)$. The size of the neighbourhood $|N(v)|$ is called the *degree* of vertex v.

A graph is called *undirected* if the edges do not have an orientation, that is, the order in which we write the endpoints u and v is irrelevant. A graph is called *simple* if it does not contain multiple edges connecting the same pair of vertices, and if it does not contain loops, i.e., edges of the form (v, v) for some vertex $v \in V$.

A graph G is called *connected* if for every pair of vertices u, v, there exists a sequence of edges connecting u to v. Otherwise, it is called *disconnected*. The maximal connected subgraphs of G are referred to as the *connected components* of G. A k-cut is a set of k vertices that, upon removing from G, increases the number of connected components of G. For $k \geq 1$, a graph is called *k-connected* if it does not have a $(k-1)$-cut. The *connectivity* of G is the smallest k such that G is k-connected.

Unless stated otherwise, all graphs in this work are simple, undirected and connected.

2.1.1 Subgraphs, subdivisions and minors

Let $G = (V, E)$ be a graph. A graph $H = (W \subseteq V, F \subseteq E)$ is called a *subgraph* of G. If H is a subgraph of G such that every edge $(u, v) \in E$ with $u, v \in W$ belongs to F as well, then H is called an *induced subgraph* of G and is denoted by $G[W]$.

An operation of replacing an edge $e = (u, v)$ with a path of length 2 from u to v is called *subdividing an edge*. A graph H obtained from G by subdividing a subset of edges with new vertices is called a *subdivision* of G. A *k-subdivision* of G is a graph in which every edge of G is subdivided precisely k times, and a *full subdivision* is a graph in which every edge of G is subdivided at least once.

Let G be a graph with an edge $e = (u, v)$. Consider the graph H obtained by removing e from G, adding a new vertex x, connecting x to all neighbours $y \in N(u) \cup N(v)$ with an edge, and subsequently removing u and v (including the incident edges). We say that H is obtained from G by *contracting edge* (u, v). If J is a subgraph of a graph obtained from G by a sequence of edge deletions and contractions, then we call J a *minor* of G. An *induced minor* is a graph H obtained from G by taking an induced subgraph and then contracting edges.

2.1.2 Common graph classes

We use the following notation for some common graph classes. A *complete graph* on n vertices is a graph that has n vertices and $\binom{n}{2}$ edges and is denoted by K_n. A *complete bipartite graph* is a graph whose vertex set V can be partitioned into two disjoint subsets A, B such that no two vertices in A and B respectively are connected by an edge, and every vertex $v \in A$ is adjacent to every vertex $w \in B$. We denote such a graph by $K_{k,\ell}$ where $k = |A|$ and $\ell = |B|$. A *path* of length k is a graph with $k + 1$ vertices where two vertices, called *ends*, have degree 1 and the rest have degree 2. A path of length k is denoted by P_k. A *cycle* of length $k \leq 3$ is a graph on k vertices where every vertex has degree 2, and is

denoted by C_k. C_3 is also called a *triangle*. A graph is called *acyclic* if it does not contain a cycle as a subgraph. A *tree* is an acyclic graph with n vertices and $n-1$ edges. A vertex of degree 1 in a tree is referred to as a *leaf*. A graph for which every connected subgraph is a tree is called a *forest*.

2.1.3 Planarity and embeddings

A *planar graph* is a graph that can be embedded in the plane, i.e., it can be drawn so that no edges intersect except at common endpoints. We assume throughout this work that planar graphs are given by a *combinatorial embedding*, i.e., by specifying the clockwise (CW) cyclic order of incident edges around each vertex. A *facial region* is a connected region of $\mathbb{R}_2 - \Gamma$ where Γ is a planar drawing of G that conforms with the combinatorial embedding. The circuit bounding this region can be read from the combinatorial embedding of G and is referred to as a *facial circuit*. We sometimes refer to both a facial circuit and a facial region as a *face* when the precise meaning is clear from the context. The *outerface* is the one that corresponds to the unbounded region; all others are called *interior faces*. The outerface cannot be read from the embedding; we assume throughout this paper that the outerface of G has been specified. An edge of G is called *interior* if it does not belong to the outerface. A vertex is called *exterior* if it is on the outerface and *interior* otherwise. A fixed combinatorial embedding of a planar graph G together with a fixed outerface is referred to as a *planar embedding* of G or, in short, a *plane graph*. We assume throughout this thesis that one planar embedding of a graph G is fixed. Subgraphs inherit this embedding, i.e., they use the induced clockwise orders. Subgraphs also inherit the outerface by using as outerface the one whose facial region contains the facial region of the outerface of G. An *outerplanar graph* is a graph that can be embedded so that all vertices are on the outerface.

The following results that characterize planar and outerplanar graphs are well known: A graph G is: (a) planar if and only if it contains no subdivision of K_5 or $K_{3,3}$ as a subgraph; and (b) outerplanar if and only if it contains no subdivision of $K_{2,3}$ or K_4.

A *maximal* planar graph is a graph where one cannot add an edge without violating planarity or simplicity. Such a graph is also often called *triangulated* because every face in its planar embedding is a triangle. Given a plane graph G, a cycle C is called *separating*

if it contains at least one vertex in its interior and at least one vertex in its exterior. A *separating triangle* is a separating cycle of length 3.

2.2 String representations

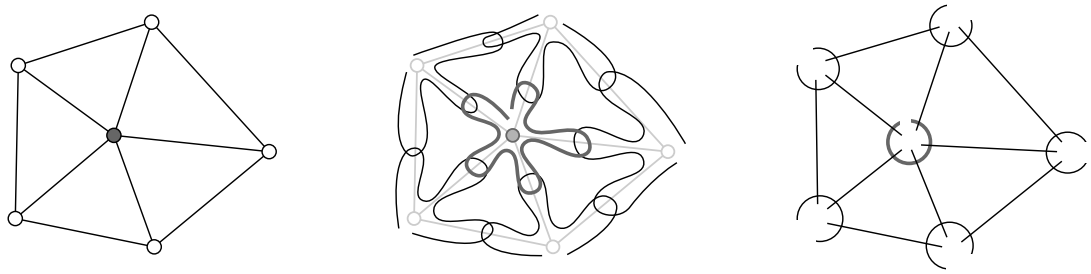
Let $G = (V, E)$ be a graph. A *string representation* R of G is a collection $R = \{\mathbf{v} \mid v \in V\}$ of curves in the plane so that $\mathbf{u} \cap \mathbf{v}$ is non-empty if and only if $(u, v) \in E$. We say that a curve $\mathbf{v}$ *represents* vertex v. In this work, we denote the curve that represents a vertex v by $\mathbf{v}$ (in bold). If the representation R is not clear from the context, we indicate it by writing $\mathbf{v}^R$.

A point that belongs to at least two curves in a representation R is called an *intersection point*. A string representation is *proper* if:

1. each $\mathbf{v}$ is a simple curve, i.e., it does not intersect itself;
2. R has finitely many intersection points;
3. each intersection point of curves belongs to precisely two curves; and
4. the cyclic order of curves $\mathbf{u}$ and $\mathbf{v}$ entering and leaving an intersection point is $\mathbf{u}, \mathbf{v}, \mathbf{u}, \mathbf{v}$.

Note that the definition of a proper representation in particular disallows two curves to overlap or touch (not even in an endpoint). Unless specified otherwise, all string representations in this thesis are assumed to be proper, and we will usually not write this qualifier. One exception is in Chapter 5, where we will consider "touching" representations where endpoints may lie on other strings, but these can easily be converted into proper string representations by extending the strings slightly[1]. Graphs that have string representations are called *string graphs*.

[1] In general, one can construct a proper string representation for any graph with a string representation that is not proper, but sometimes two proper intersections between two curves need to be created in order to replace a contact.



(a) A planar graph. (b) A string representation. (c) A string representation of a full subdivision.

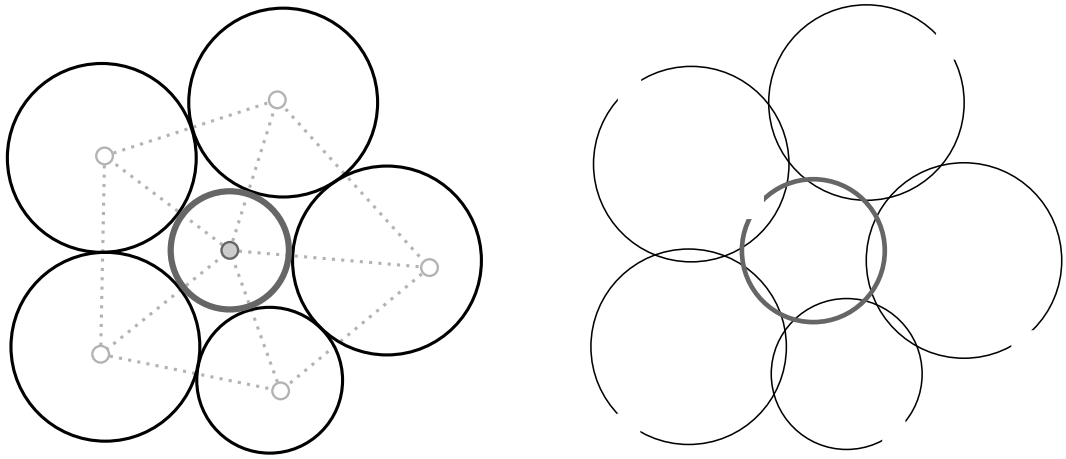
Figure 2.1: Constructing string representation for planar graphs by tracing along edges [40].

String graphs were first studied in 1976 in the paper of Ehrlich, Even and Tarjan [40], but similar concepts appeared before in the work of Benzer [10] and Sinden [80]. Ehrlich, Even and Tarjan showed that every planar graph has a string representation [40]. They noted that, given a planar drawing of a graph G, one can create a curve $\mathbf{v}$ for every vertex v so that $\mathbf{v}$ traces (see Figure 2.1(b)) the edges incident to v to just beyond one half of their length. This way, the curves $\mathbf{u}$ and $\mathbf{v}$ intersect whenever an edge (u, v) is present. See also Figure 2.1.

Another easy argument showing the existence of string representations of planar graphs uses the theorem that every planar can be represented by touching circles in the plane [66]. This is a so-called *circle packing representation*. Having a circle packing representation of a planar graph G, one can enlarge every circle by a small ε, forcing the circles to intersect. Since every circle can be turned into a curve by breaking at a suitable point, a string representation exists. See Figure 2.2.

Given the existence of string representations for planar graphs, one immediately wonders whether all graphs are in fact string graphs. It is well known that this is not the case. We review the proof here because we will use similar arguments later.

We first need to define the following operation. Let $\mathbf{v}$ be a curve that represents a vertex v of degree 2, and let x, y be the neighbours of v. The operation of *contracting*



(a) A circle packing representation corresponding to the graph from Figure 2.1.

(b) A matching string representation.

Figure 2.2: Constructing string representations from circle packing representations.

curves $\mathbf{x}$ and $\mathbf{y}$ means choosing and endpoint of each $\mathbf{x}$ and $\mathbf{y}$, placing vertices x and y into those endpoints, and replacing $\mathbf{v}$ with a curve $\mathbf{v}'$ that starts in x, follows $\mathbf{x}$ into its last intersection with $\mathbf{v}$ before intersecting $\mathbf{y}$, then follows $\mathbf{v}$ into its first intersection with $\mathbf{y}$, and follows $\mathbf{y}$ into y where it ends. Note that resulting curve $\mathbf{v}'$ is not crossed by any other curve.

Lemma 2.1 (Kratochvíl [67])**.** *A full subdivision of a non-planar graph is not a string graph.*

Proof. Let G be a non-planar graph, and let H be a full subdivision. Assume that a string representation of H exists. Contract every curve v for a vertex $v \in V(G)$ into a point. Observe that since every vertex $u \in V(H) \setminus V(G)$ has degree 2, after the contraction, no curve in the representation is crossed. Thus, one can place the vertices of G in the contraction points and, for every edge in G, find a non-crossed sequence of curve segments that connects its endpoints. Thus, the representation of G contains a planar drawing of G, which is a contradiction. □

On the other hand, if H is a full subdivision of a planar graph G, then a string

representation of H is easily constructed from a planar drawing of G by replacing every vertex by a short string. See Figure 2.1c.

Recall that an *induced minor* is a graph H obtained from G by taking an induced subgraph and then contracting edges. Kratochvíl argues that string graphs are closed under taking induced minors [67]. We briefly review this here since it often will be crucial. First, if $G[W]$ is an induced subgraph of a string graph G, then the subset of curves $\mathbf{w}$ for every $w \in W$ in any string representation of G forms a string representation of $G[W]$. Thus, we have the following:

Corollary 2.2. *Let G be a graph that contains a full subdivision of a non-planar graph H as an induced subgraph. Then G is not a string graph.*

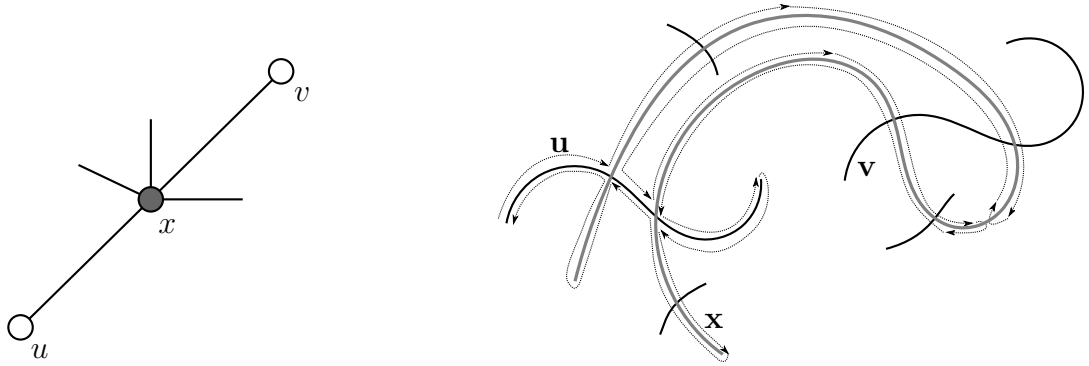
Observation 2.3. *If G is a string graph, and H is obtained from G by contracting an edge, then H is a string graph.*

Proof. Let G be a string graph and let H be obtained from G by contracting an edge (u, x). One can use a string representation of G to produce a string representation of H by tracing along $\mathbf{x}$ with curve $\mathbf{u}$ (see Figure 2.3). Formally, if $\mathbf{x}$ intersects $\mathbf{u}$ multiple times, break $\mathbf{x}$ into segments so that $\mathbf{u} \cup \mathbf{x}$ does not contain cycles. Then, for a sufficiently small $\varepsilon > 0$, consider the set of points with distance at most ε to $\mathbf{u} \cup \mathbf{x}$. The boundary of this area is a closed curve that intersects all the curves intersected by $\mathbf{u} \cup \mathbf{x}$. Breaking the boundary in a point turns it into a string that replaces $\mathbf{x}$ and $\mathbf{u}$ after contracting (u, x). □

The complexity of recognizing string graphs will be discussed in Section 2.2.4.

2.2.1 1-string representations

Both the construction of string representations for planar graphs from [40] and the one based on circle-packing representation have the property that some curves intersect each other twice. However, one would naturally want the curves to be more "well-behaved" and intersect each other only once. Such representations are called 1-*string representations*. A k-*string* representation is a generalization of a 1-string representation where every two



(a) An edge (u, x) to be contracted.

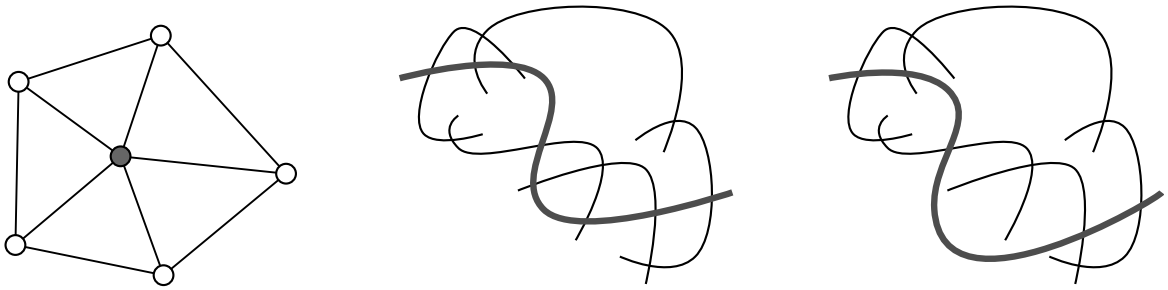
(b) Re-routing curve **u** to trace along **x**.

Figure 2.3: Illustration for the proof of Observation 2.3 and constructing a string representation for a graph when contracting an edge.

curves are allowed to intersect each other at most k times. The classes of graphs with 1-string and k-string representations are called 1-String and k-String, respectively. See also Figure 2.4.

One could restrict string representations even further by requiring that the curves have some specific shape. In 1984, Scheinerman conjectured that all planar graphs can be represented as intersection graphs of line segments [79]. Such representations are called *segment representations* and the corresponding class of graphs is Seg. See also Figure 2.5c. Note that any graph in Seg is in 1-String.

We first list some graph classes that clearly belong to Seg. One can immediately see that segment representations exist for cliques. A complete bipartite graph $K_{m,n}$ can be represented by m horizontal segments intersecting n vertical segments. A *circle graph* is an intersection graph of chords in a circle, thus it has segment representation by definition. A graph G with vertices $\{1, \ldots, n\}$ is called a *permutation graph* if there exist two permutations π_1, π_2 of $\{1, \ldots, n\}$ such that (i, j) is an edge of G if and only if π_1 lists i, j in the opposite order as π_2 does. Put differently, if we place $\pi_1(1), \ldots, \pi_1(n)$ at points along a horizontal line, and $\pi_2(1), \ldots, \pi_2(n)$ at points along a parallel horizontal line, and use the line segment $(\pi_1(i), \pi_2(i))$ to represent vertex i, then the graph is the intersection graph of these segments.



(a) A planar graph. (b) A string representation. (c) A 1-string representation.

Figure 2.4: A planar graph with a string and 1-string representation.

Again, by definition, permutation graphs have segment representations.

Scheinerman's conjecture was proved first for bipartite planar graphs [38, 59] with the strengthening that every segment is vertical or horizontal. We review this result in Section 5.1.1. Later, the conjecture was proved also for planar triangle-free graphs, which can be represented by line segments with at most three distinct slopes [36].

As a crucial step towards proving Scheinerman's conjecture, Chalopin, Gonçalves and Ochem showed in 2007 that every planar graph is in 1-String [30, 31]. We will review the idea of this construction, and build on top of it for a stronger result in Chapter 3.

Scheinerman's conjecture was finally resolved in 2009 when it was proved by Chalopin and Gonçalves [29] by extending the techniques of their previous result [30, 31].

Note that the construction referenced in Observation 2.3 creates curves that intersect multiple times and thus Observation 2.3 does not hold for 1-string graphs.

2.2.2 B_k-VPG representations

Recall that one motivation for studying string graphs was connections in circuit boards. Such connections usually consist of linear segments with restricted angles, and usually are seen as *orthogonal curves*, i.e., curves that consist of horizontal and vertical segments only. Such curves can be embedded as paths in a rectangular grid. We hence focus on graphs with

such a string representation and call a graph a *VPG graph*[2] if it has a string representation that uses only orthogonal curves. The class of VPG graphs was introduced by Asinowski et al. [8].[3] The *B_k-VPG graphs* are the graphs that have a string representation where curves are orthogonal and have at most k bends. See also Figure 2.5b.

It is easy to see that all string graphs are VPG graphs as any string representation can be embedded into a rectangular grid with a sufficient resolution. In particular, it follows that planar graphs are VPG graphs and a VPG representation can be obtained, e.g., from the construction of Ehrlich, Even and Tarjan. For bipartite planar graphs, orthogonal curves can even be required to have no bends [38, 59] (see also Section 5.1.1) so they are in B_0-VPG. For arbitrary planar graphs, bends are required in orthogonal curves. Chaplick and Ueckerdt showed that 2 bends per curve always suffice [35], i.e., that planar graphs are in B_2-VPG. Unfortunately, in Chaplick and Ueckerdt's construction, curves may cross each other repeatedly, and so it does not prove that planar graphs are in 1-String. In Chapter 3, we strengthen their results and prove that planar graphs have string representations that are simultaneously 1-string and B_2-VPG.

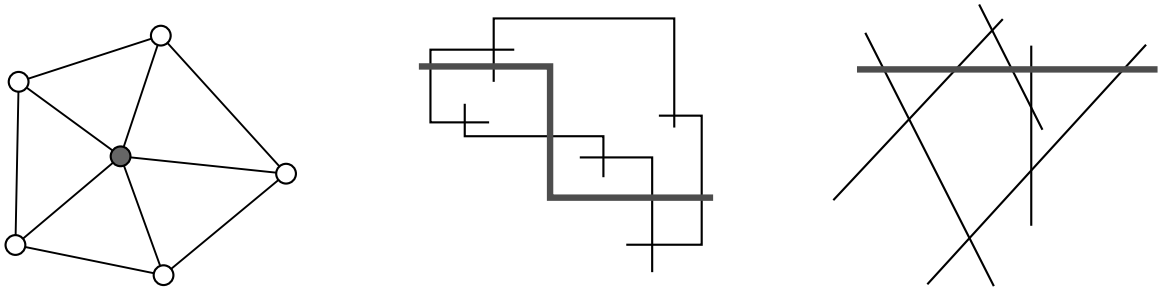
One advantage of B_k-VPG representations is that the coordinates needed to describe such a representation are small, a result that will be useful later:

Lemma 2.4. *For any B_k-VPG representation R, there is a B_k-VPG representation R' such that all segments have distinct coordinates in an $O(kn) \times O(kn)$-grid.*

Proof. Every vertical segment s in R is intersected by horizontal segments only. As all the intersections are proper, s can be shifted by a small amount both left and right. By repeatedly shifting a vertical segment, we can construct a representation in which all vertical segments have distinct coordinates. Subsequently, we can apply an analogous argument to horizontal segments, and repeatedly shifting a segment up or down, we can construct a representation R' in which all segments have distinct coordinates.

[2] As in "Vertex-intersection graph of Paths in a Grid."

[3] The definition of VPG graphs used here is slightly different from [8]. While in [8], two paths in a grid intersect whenever they share a point (e.g., overlaps count as intersections), we require that all crossing are proper (and disallow overlaps).



(a) A planar graph. (b) A B_2-VPG representation. (c) A segment representation.

Figure 2.5: A planar graph with a B_2-VPG and segment representation.

Since every curve has at most k bends, it has at most $k+1$ segments. The entire representation has at most $n(k+1)$ segments all together. Since only the coordinates that contain a segment are necessary, the entire representation can be embedded into a grid of size $O(kn) \times O(kn)$. □

B_k-VPG representations were further investigated by Chaplick, Jelínek, Kratochvíl and Vyskočil [67] who studied the complexity of recognition (see more details in Section 2.2.4) and the relationship between B_k-VPG and B_{k+1}-VPG graphs (see Sections 2.2.4 and 5.3).

Apart from the motivation of B_k-VPG graphs mentioned above, we are especially interested in B_1-VPG graphs, and in particular those that use only ⌞ and ⌜ as shapes. The reason is that there is a relationship between such graphs (which we call {⌞,⌜}-graphs) and SEG: every such B_1-VPG representation can be "stretched" into a segment representation. This statement was proved by Middendorf and Pfeiffer in 1992 [74]. In fact, a stronger claim holds: The stretching preserves the order of intersections along curves. We state the result here, and give the proof in Chapter 6 which will be concerned with *order-preserving* (in some sense) string representations.

Lemma 2.5 (Middendorf, Pfeiffer [74])**.** *Let G be a graph with a B_1-VPG representation R that uses only curves of shapes ⌞ and ⌜. There is a string representation S of G such that every curve in S is a line segment.*

Naturally, one wonders whether the other direction is true. Thus, can every segment representation be converted to a B_1-VPG representation with $\llcorner$ and $\ulcorner$ curves? To our knowledge, this question remains open. Note that if it were true, then every planar graph would be a B_1-VPG graph by the results of Chalopin and Gonçalves [29]. As it is, we do not know whether every planar graph is a B_1-VPG graph, and this is, in fact, one of the big remaining open questions in the field[4].

2.2.3 Outer-string representations

An *outer-string representation* of a graph is a string representation in which curves lie inside a disk such that each curve attaches to the boundary of the disk by one of its endpoints. An *outer-string graph* is a graph that has an outer-string representation. Outer-string graphs were introduced by Kratochvíl in 1991 [67] and investigated later, see e.g. [48, 47]. Clearly, not all graphs are outer-string by Lemma 2.1. However, even some string graphs are not outer-string.

Lemma 2.6. *Let G be a planar graph that is not outer-planar. Any full subdivision H of G is a string graph that does not have an outer-string representation.*

Proof. Since G is planar, so is H, and thus it has a string representation. The proof that H cannot be outer-string is similar to the proof of Lemma 2.1. Assume that an outer-string representation R of H exists. Let B be the boundary of R. Contract every curve $\mathbf{v}$ for a vertex $v \in V(G)$ into its end on B. Observe that since every vertex $u \in V(H) \setminus V(G)$ has degree 2, after the contraction, no curve in the representation is crossed. Thus, the representation is planar. One can place the vertices of G at the contraction points and, for every edge in G, find a non-crossed sequence of curve segments that connects its endpoints. Since all the curves lie inside B and all the contracted points lie on B, the representation contains an outer-planar drawing of G, which is a contradiction. □

[4]Very recently, after this thesis has been submitted, Gonçalves et al. [55] positively resolved the question by proving that every planar graph has a representation using intersecting $\llcorner$'s.

Similarly to string graphs, one can pose some restrictions to the shapes of curves in an outer-string representation and derive subclasses of outer-string graphs such as 1-outer-string, outer-segment [43], and B_k-outer-VPG graphs.

A comprehensive study of outer-string graphs was done by Cabello and Jejčič [25]. They showed that for any outer-planar graph G, any full subdivision H is an outer-string graph which is also an outer-segment graph and a circle graph. Furthermore, this is an "if and only if" relationship. This raises the question of whether all outer-string graphs are outer-segment graphs. The answer to this, from the very same paper, is negative: there are outer-string graphs that are not segment graphs, and thus they are not outer-segment. (We will prove an even stronger statement in Chapter 7, where we argue that in outer-string representations, curves sometimes have to cross each other an exponential number of times.) On the other hand, there are graphs that are segment graphs but not outer-string graphs. Hence, there is no containment between SEG and outer-string graphs [25]. Note that the classes of outer-string and outer-segment graphs also are not subclasses of outer-planar and planar graphs as every complete graph K_n is an outer-segment graph.

2.2.4 Complexity of recognition

String graph recognition was proved to be NP-hard by Kratchovíl in 1991 by a reduction from AT-REALIZABILITY [68]. We briefly review some key ideas here.

An *abstract topological graph*, in short an *AT-graph*, is a triple $(V(G), E(G), I)$ where $I \subseteq \binom{E(G)}{2}$. AT-REALIZABILITY is the problem where, given an abstract topological graph $(V(G), E(G), I)$, we ask whether there is a drawing of graph G in the plane where two edges e, f cross if and only if $\{e, f\} \in I$. We say that a drawing *realizes* the AT-graph. Kratochvíl proved that this problem is NP-hard [68] by reduction from planar 3-connected 3-SAT. Then he showed:

Lemma 2.7 (Kratochvíl [68])**.** AT-REALIZABILITY *reduces to the problem of string graph recognition.*

Proof. Given an AT-graph $G = (V, E, I)$, define:

- a set of incidence-vertices $X = \{(u, e) | e \in E \text{ and } u \text{ is incident to } e\}$
- for every incidence-vertex $x = (u, e) \in X$, two incidence-edges (u, x) and (e, x). Let Y be all these incidence-edges.

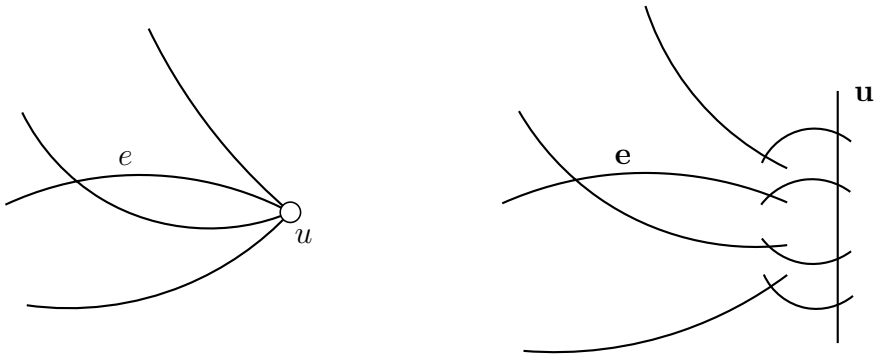
Consider the graph $H = (W = V \cup E \cup X, F = Y \cup I)$. We claim that H is a string graph if and only if G is realizable.

Let G be realized as a drawing Γ. Obtain a string representation of G by shortening every edge e slightly to obtain $\mathbf{e}$, replacing each vertex v of Γ with a short curve to get $\mathbf{v}$, and connecting $\mathbf{v}$ to an incident $\mathbf{e}$ with a curve that represents vertex $(v, e) \in X$. See Figure 2.6.

For the converse, let H be a string graph. Since the vertices in X have degree 2, we may assume that every curve $\mathbf{x}$ has precisely two intersections with other curves [68, p. 68]. Let $e = (v, w)$ be an edge of G, and let $x_v = (e, v)$ and $x_w = (w, e)$ be its incidence vertices. By walking around $\mathbf{e}$ and cutting off appropriately, we may assume that $\mathbf{e}$ begins and ends at its intersections with $\mathbf{x_v}$ and $\mathbf{x_w}$, see also Figure 2.7. By contracting every curve $\mathbf{v}$ for $v \in V$ and $\mathbf{x}$ for a vertex $x \in X$ a point, we obtain a drawing of G where edges e and e' intersect if and only if $\mathbf{e}$ and $\mathbf{e}'$ intersected, which implies that $(e, e') \in I$. □

In consequence, recognizing string graphs is NP-hard. Note that the proof of Lemma 2.7 does not imply hardness of recognition of 1-string graphs as At-Realizability does not pose any restrictions on the number of intersections between two edges. Also, the string representation used to produce a realization in the proof may require multiple intersections of two curves as edge-curves need to start in the proximity of incidence-curves (cf. Figure 2.7). NP-hardness of recognizing 1-string graphs was proved, with an entirely different approach, in [69].

It is not straightforward to see whether recognizing string graphs belongs to NP: there are string graphs that have string representations that require a number of crossings exponential in n ([70], we review this in Section 7.1). Thus, string representations cannot be simply "non-deterministically guessed" in polynomial time. The proof that the recognition problem of string graphs is in NP was provided by Schaefer et al. in 2003 [78]. So we have:



(a) (b)

Figure 2.6: (a) A neighbourhood of a vertex in a drawing of a graph. (b) A corresponding representation in a string graph.

(a) Original position of **e**.

(b) Bending and routing **e** so that it ends by the incidence-vertex curve.

Figure 2.7: An edge curve can be extended so that it has both ends in the proximity of an incidence-vertex curve.

Lemma 2.8 (Schaefer et al. [78])**.** *The recognition problem of string graphs is NP-complete.*

We now briefly discuss complexity issues for string representations with restricted shapes:

- The recognition problem of graphs with segment representations (class SEG) is complete in the existential theory of the reals, $\exists\mathbb{R}$ [69]. The complexity class $\exists\mathbb{R}$ is known to contain NP and is known to belong to PSPACE [77]. However, it is unknown if $\exists\mathbb{R}$ belongs to NP. Thus, we do not know if recognizing segment graphs is in NP.

- The recognition of k-string graphs is in NP due to the bound on the number of intersections. For each curve, one can non-deterministically "guess" the number and order of intersections with other curves, which provides a unique combinatorial description of the representation. Testing whether this can be realized then amounts to testing planarity of a suitably defined graph (see also Lemma 6.1 in Chapter 6 where we review this in more detail).

  For every k, the recognition problem of k-string graphs was established to be NP-complete by Kratochvíl in 1994 [69]. By the same paper, the recognition of 1-string graphs is NP-hard even if a 2-string representation of the same graph is given. Also, for every k, k-STRING $\neq k+1$-STRING.

- The situation is much simpler for B_k-VPG graphs as their representations can be embedded into a grid of size $O(kn) \times O(kn)$. Thus, for a fixed k, the recognition problem of B_k-VPG graphs is clearly in NP. The hardness was shown by Chaplick et al. in 2012 [33] who showed that for every k, the class of B_{k+1}-VPG graphs is strictly larger than B_k-VPG graphs, and the recognition problem of B_k-VPG graphs is NP-complete even if a B_{k+1}-VPG representation is given.

- The complexity of recognition of outer-string graphs is open. We will show in Section 7.2, Lemma 7.2 that it is in NP, but it remains unknown whether it is NP-hard.

2.2.5 Algorithmic implications of string representations

There are a number of algorithmic results for various classes of string and outer-string graphs, which we list here. The bottom line is that imposing restrictions on curves in a string representation might be favourable for algorithmic questions.

We first list some results for arbitrary string graphs. The Maximum Clique problem on string graphs was investigated by Middendorf and Pfeiffer [74]. They showed that it remains NP-hard in general string graphs, but presented polynomial algorithms for some restricted subclasses of string graphs (so-called *opposite angle string graphs*). The *cop number* of a graph G is the smallest k such that k cops win the game of cops and robber on G. Gavenčiak et al. [51] showed that string graphs have cop number at most 15.

There also exist some results that utilize a divide-and-conquer approach. We need a definition. A *separator* in a graph $G = (V, E)$ is a subset S of the vertex set V such that no connected component of $G[V \setminus S]$ has more than $\frac{2}{3}|V|$ vertices. Matoušek [73] showed that every string graph with m edges admits a vertex separator of size $O(\sqrt{m}\log m)$. Fox and Pach conjectured that every string graph has a separator of size $O(\sqrt{m})$ [45]. This has been proved for k-string graphs if k is constant [44] and very recently for all string graphs by Lee in [72]. Algorithmic consequences of separators in string graphs are discussed in [45, 49]. One example of a result based on separators is an n^{ε}-approximation algorithm for Maximum Independent Set in k-string graphs by Fox and Pach [46]. Har-Peled and Quanrund [58] show that separator theorems are applicable for approximation algorithms in all sparse string graphs. However, none of results seem to lead to approximation algorithms with factors better than $O(n^{\varepsilon})$ for all string graphs.

Now we list some results for outer-string graphs. In 2015, Keil et al. [63, 61] described an algorithm based on dynamic programming for the Maximum Weight Independent Set problem in an outer-string graph that runs in time polynomial in the size of the geometric input representation of the graph. This is an especially interesting result since Minimum Clique Cover, Colorability, Minimum Dominating Set, and Hamiltonian Cycle are NP-complete for outer-string graphs as they contain circle graphs, the class of intersection graphs of chords in a circle, as a subclass. Rok and Walczak [76] proved that the number of colours needed for an outer-string graph G is a function of the maximum clique size $\omega(G)$

(the graphs are so-called χ-bounded).

Finally, we list some results for B_k-VPG graphs. Since these include planar graphs for $k \geq 2$, most problems remain NP-hard on B_k-VPG graphs. Lahiri et al. gave an $O(\log n)$-approximation algorithm for independent set in B_1-VPG graphs [71] (we will build on top of this in Chapter 4). We know of no other algorithmic results for B_k-VPG graphs, though many results are known for so-called B_k-EPG graphs (see e.g. [20, 21] and the references therein for the definition and more details).

2.3 Organization of the thesis

In this thesis, we investigate string representations with emphasis on the shapes of curves and the way they intersect. Refer to Figure 2.8 which visualizes the relationships between classes of string representations. The thesis is organized as follows. Having reviewed definitions and some known results in this chapter, in Chapter 3 we explore B_k-VPG representations of planar graphs. Recall that the class of planar graphs has been proved to lie inside SEG, but it is an open question whether all planar graphs have B_1-VPG representations[5]. We strengthen some previously known results and show that B_2-VPG representations that are simultaneously 1-string exist for planar graphs. In Chapter 4, we show that with some restrictions on the shapes of the curves, string representations can be used to produce approximation algorithms for several problems. The B_2-VPG representations constructed in Chapter 3 satisfy these restrictions. In Chapter 5, we attempt to further restrict the number of bends in VPG representations for subclasses of planar graphs, and investigate B_1-VPG representations, especially for planar partial 3-trees and some subclasses of them. In Chapter 6, we propose new classes of string representations for planar graphs that we call "order-preserving." Order-preservation is an interesting property which relates the string representation to the planar embedding of the graph, and we believe that it might prove useful when constructing string representations. In Chapter 7, we turn towards graphs that are not 1-string and not even k-string for any polynomial k. We show that there are outer-string graphs that require more than an exponential number of crossings in their

[5]See Section 8.2 for the discussion of recent results of Gonçalves et al. [55].

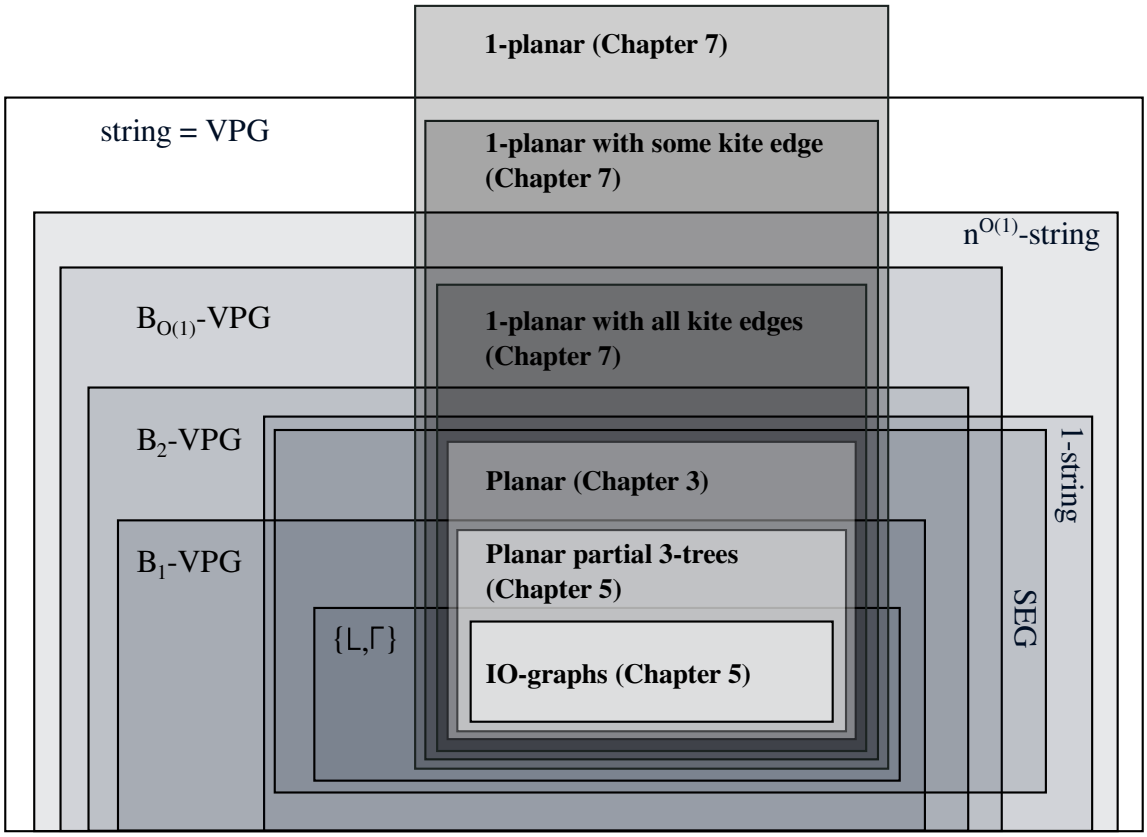


Figure 2.8: The relationships between classes of string representations.

outer-string representations. Our construction will also prove that so-called 1-planar graphs are not always 1-string, but under some restrictions, they have B_k-VPG representations for some constant k. We conclude in Chapter 8.

Chapter 3

String Representations of Planar Graphs

In this chapter, we show that every planar graph has a string representation that simultaneously satisfies the requirements for 1-String (any two curves cross at most once) and the requirements for B_2-VPG (any curve is orthogonal and has at most two bends). Our result hence re-proves, in one construction, the results by Chalopin et al. [30, 31] and the result by Chaplick and Ueckerdt [35].

Theorem 3.1. *Every planar graph has a* 1*-string* B_2*-VPG representation.*

Our construction for the proof of Theorem 3.1 will use all 8 possible shapes of B_2-VPG curves. As mentioned in Section 2.2, an advantage of B_k-VPG representations is that the coordinates to describe such a representation are small—orthogonal drawings can be deformed easily such that all bends are at integer coordinates. Every vertex curve has at most two bends and hence at most 3 segments, so the representation can be made to have coordinates in an $O(n) \times O(n)$-grid with perimeter at most $3n$. Note that none of the previous results provided an intuition of the required size of the grid.

In addition to Theorem 3.1, we show that for 4-connected planar graphs, only a subset of orthogonal curves with 2 bends is needed:

Theorem 3.2. *Every* 4*-connected planar graph has a* 1*-string* B_2*-VPG representation where all curves have a shape of* ⌊, ⌉, ⌋ *or* ⌈.

Our approach is inspired by the construction of 1-string representations by Chalopin, Gonçalves and Ochem from 2007 [30, 31]. The authors proved the result in two steps. First, they showed that maximal planar graphs without separating triangles admit 1-string representations. By induction on the number of separating triangles, they then showed that a 1-string representation exists for any maximal planar graph, and consequently for any planar graph.

In order to show that maximal planar graphs without separating triangles have 1-string representations, Chalopin et al. [31] used a method inspired by Whitney's proof that 4-connected planar graphs are Hamiltonian [85]. Asano, Saito and Kikuchi later improved Whitney's technique and simplified his proof [7]. We use the same approach as [31], but borrow ideas from [7] and develop them further to reduce the number of cases. Even so, the proof is quite complicated. The reader may wish to consult Section 3.5, where we illustrate the construction on a small graph.

The results of this chapter appeared in [12, 13].

3.1 Definitions and basic results

Let us restate a formal definition of a 1-*string* B_2*-VPG representation*.

Definition 3.3 (1-string B_2-VPG representation)**.** *A graph* G *has a* 1-string B_2-VPG representation *if every vertex* v *of* G *can be represented by a curve* $\mathbf{v}$ *such that:*

1. *Curve* $\mathbf{v}$ *is* orthogonal*, i.e., it consists of horizontal and vertical segments.*
2. *Curve* $\mathbf{v}$ *has at most two bends.*
3. *Curves* $\mathbf{u}$ *and* $\mathbf{v}$ *intersect at most once, and* $\mathbf{u}$ *intersects* $\mathbf{v}$ *if and only if* (u, v) *is an edge of* G*.*

In this chapter, "representation" means "1-string B_2-VPG representation" since we do not consider any other representations.

Our technique for constructing 1-string B_2-VPG representations of a graph uses an intermediate step referred to as a "partial 1-string B_2-VPG representation of a W-triangulation that satisfies the chord condition with respect to three chosen corners." We define these terms, and related graph terms, first.

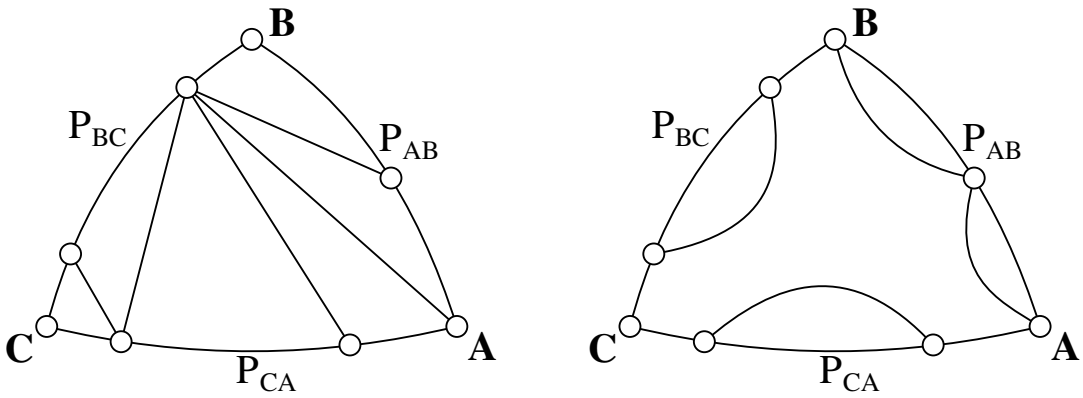
A *triangulated disk* is a planar graph G for which the outerface is a simple cycle and every interior face is a triangle. Recall that a *separating triangle* is a cycle C of length 3 such that G has vertices both inside and outside the region bounded by C (with respect to the fixed embedding and outerface of G). Following the notation of [31], a *W-triangulation* is a triangulated disk that does not contain a separating triangle. Recall that a *chord* of a triangulated disk is an interior edge for which both endpoints are on the outerface.

Let X, Y be two vertices on the outerface of a connected planar graph so that neither of them is a cut vertex. Define P_{XY} to be the counter-clockwise (CCW) path on the outerface from X to Y (including X and Y). We often study triangulated disks with three specified distinct vertices A, B, C called the *corners*. A, B, C must appear on the outerface in CCW order. We denote $P_{AB} = (a_1, a_2, \ldots, a_r)$, $P_{BC} = (b_1, b_2, \ldots, b_s)$ and $P_{CA} = (c_1, c_2, \ldots, c_t)$, where $c_t = a_1 = A$, $a_r = b_1 = B$ and $b_s = c_1 = C$.

Definition 3.4 (Chord condition)**.** *A W-triangulation G satisfies the* chord condition *with respect to the corners A, B, C if G has no chord within P_{AB}, P_{BC} or P_{CA}, i.e., no interior edge of G has both ends on P_{AB}, or both ends on P_{BC}, or both ends on P_{CA}.*[1]

Definition 3.5 (Partial 1-string B_2-VPG representation)**.** *Let G be a connected planar graph and $E' \subseteq E(G)$ be a set of edges. An* (E')-1-string B_2-VPG representation *of G is a 1-string B_2-VPG representation of the subgraph $(V(G), E')$, i.e., curves $\mathbf{u}, \mathbf{v}$ cross if and*

[1]For readers familiar with [31] or [7]: A W-triangulation that satisfies the chord condition with respect to corners A, B, C is called a *W-triangulation with 3-boundary P_{AB}, P_{BC}, P_{CA}* in [31], and the chord condition is the same as *Condition (W2b)* in [7]. Also, for readers familiar with Tutte's planar graph drawing results [82, 83], satisfying the chord condition is the same (for internally triangulated graphs) as having a drawing with all outer-face vertices on a triangle and A,B,C at the corners of the triangle.



(a) The chord condition is satisfied.

(b) The chord condition is violated.

Figure 3.1: An illustration of a W-triangulation with chords that do and do not satisfy the chord condition.

only if (u, v) *is an edge in* E'*. If* E' *consists of all interior edges of* G *as well as some set of edges* F *on the outerface, then we write* $(int \cup F)$ representation *instead.*

In our constructions, we use $(int \cup F)$ representations with $F = \emptyset$ or $F = \{e\}$, where e is an outerface edge incident to corner C of a W-triangulation. Edge e is called the *special edge*, and we sometimes write $(int \cup e)$ representation, rather than $(int \cup \{e\})$ representation.

3.1.1 Representation layouts

To create 1-string representations where vertex-curves have few bends, we need to impose geometric restrictions on representations of subgraphs. Unfortunately, no one type of layout seems sufficient for all cases, and we will hence have three different layout types illustrated in Figure 3.2. We will be using the layouts to construct representations of W-triangulations, however, we define them for 2-connected graphs in general.

Definition 3.6 (2-sided layout)**.** *Let* G *be a connected planar graph and* A, B *be two distinct outerface vertices such that* $G \cup \{(A, B)\}$ *is 2-connected. An* $(int \cup F)$ *1-string* B_2*-VPG representation of* G *(for some set* F*) has a* 2-sided layout *(with respect to corners* A, B*) if:*

1. *There exists a rectangle Θ that contains all intersections of curves and such that*

   (i) *the top of Θ is intersected, from right to left in order, by the curves of the vertices of P_{AB},*

   (ii) *the bottom of Θ is intersected, from left to right in order, by the curves of the vertices of P_{BA}.*

2. *Any curve $\mathbf{v}$ of an outerface vertex v has at most one bend. (By (1.), this implies that $\mathbf{A}$ and $\mathbf{B}$ have no bends.)*

Definition 3.7 (3-sided layout)**.** *Let G be a W-triangulation and A, B, C be three distinct vertices in CCW order on the outerface of G. Let F be a set of exactly one outerface edge incident to C. An $(int \cup F)$ 1-string B_2-VPG representation of G has a* 3-sided layout *(with respect to corners A, B, C) if:*

1. *There exists a rectangle Θ containing all intersections of curves so that*

   (i) *the top of Θ is intersected, from right to left in order, by the curves of the vertices on P_{AB};*

   (ii) *the left side of Θ is intersected, from top to bottom in order, by the curves of the vertices on $P_{Bb_{s-1}}$, possibly followed by $\mathbf{C}$;* [2]

   (iii) *the bottom of Θ is intersected, from right to left in order, by the curves of vertices on P_{c_2A} in reverse order, possibly followed by $\mathbf{C}$;*[2]

   (iv) *curve $\mathbf{b_s} = \mathbf{C} = \mathbf{c_1}$ intersects the boundary of Θ exactly once; it is the bottommost curve to intersect the left side of Θ if the special edge in F is (C, c_2), and $\mathbf{C}$ is the leftmost curve to intersect the bottom of Θ if the special edge in F is (C, b_{s-1}).*

2. *Any curve $\mathbf{v}$ of an outerface vertex v has at most one bend. (By (1.), this implies that $\mathbf{B}$ has precisely one bend.)*

3. *$\mathbf{A}$ and $\mathbf{C}$ have no bends.*

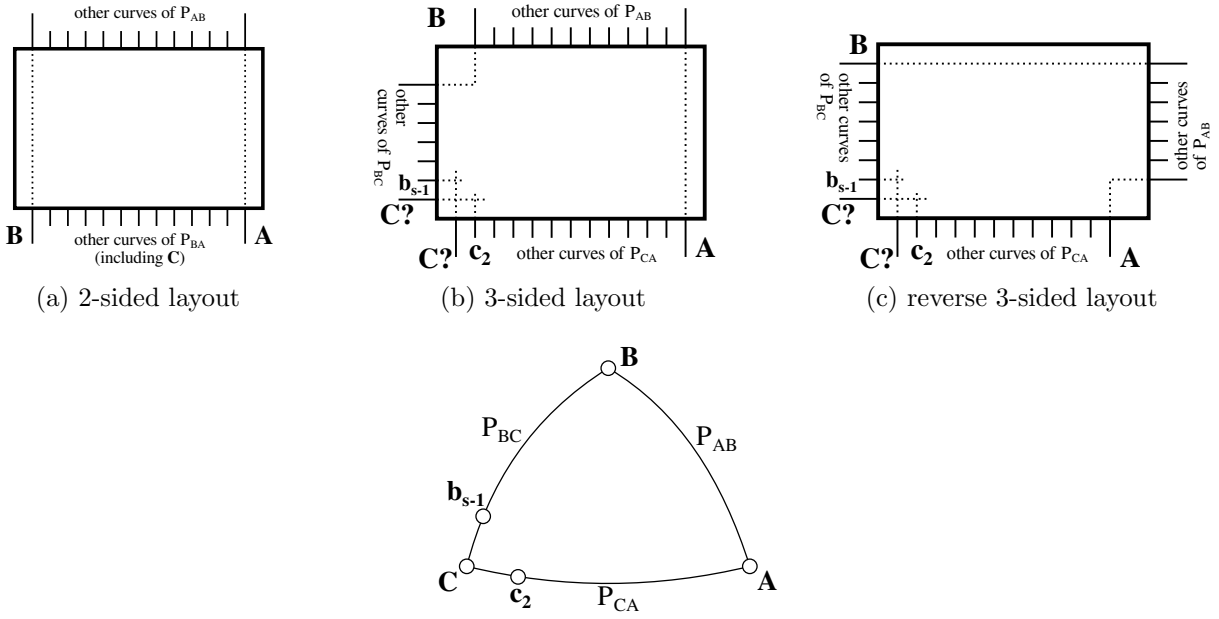
[2] Recall that (b_{s-1}, C) and (C, c_2) are the two incident edges of C on the outerface.

See Figures 3.2a and 3.2b for illustrations of a 2-sided and 3-sided layout. We also need the concept of a *reverse* 3*-sided layout*, which is similar to the 3-sided layout except that B is straight and A has a bend (see Figure 3.2c). Formally:

Definition 3.8 (Reverse 3-sided layout)**.** *Let G be a W-triangulation and A, B, C be three distinct vertices in CCW order on the outerface of G. Let F be a set of exactly one outerface edge incident to C. An ($int \cup F$) 1-string B_2-VPG representation of G has a* reverse 3-sided layout *(with respect to corners A, B, C) if:*

1. *There exists a rectangle Θ containing all intersections of curves so that*
   - *(i) the right side of Θ is intersected, from bottom to top in order, by the curves of the vertices on P_{AB};*
   - *(ii) the left side of Θ is intersected, from top to bottom in order, by the curves of the vertices on $P_{Bb_{s-1}}$, possibly followed by $\mathbf{C}$;*
   - *(iii) the bottom of Θ is intersected, from right to left in order, by the curves of vertices on P_{c_2A} in reverse order, possibly followed by $\mathbf{C}$;*
   - *(iv) curve $\mathbf{b_s} = \mathbf{C} = \mathbf{c_1}$ intersects the boundary of Θ exactly once; it is the bottommost curve to intersect the left side of Θ if the special edge in F is (C, c_2), and $\mathbf{C}$ is the leftmost curve to intersect the bottom of Θ if the special edge in F is (C, b_{s-1}).*
2. *Any curve $\mathbf{v}$ of an outerface vertex v has at most one bend. (By 1., this implies that $\mathbf{A}$ has precisely one bend.)*
3. *$\mathbf{B}$ and $\mathbf{C}$ have no bends.*

We sometimes refer to the rectangle Θ for these representations as a *bounding box*. Figure 3.3a (which will serve as base case later) shows such layouts for a triangle and varying choices of F.



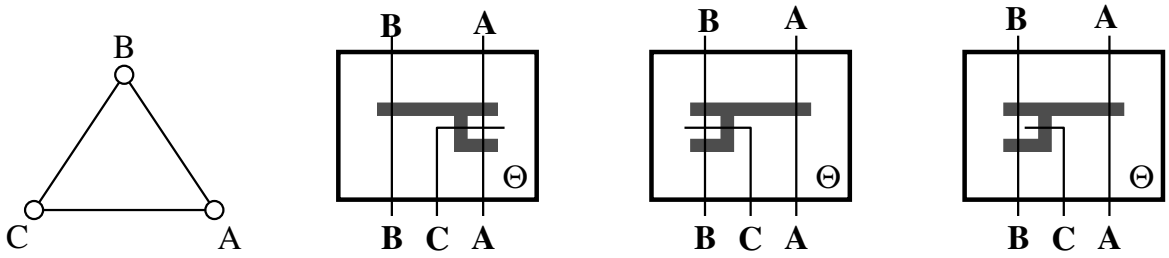
(a) 2-sided layout

(b) 3-sided layout

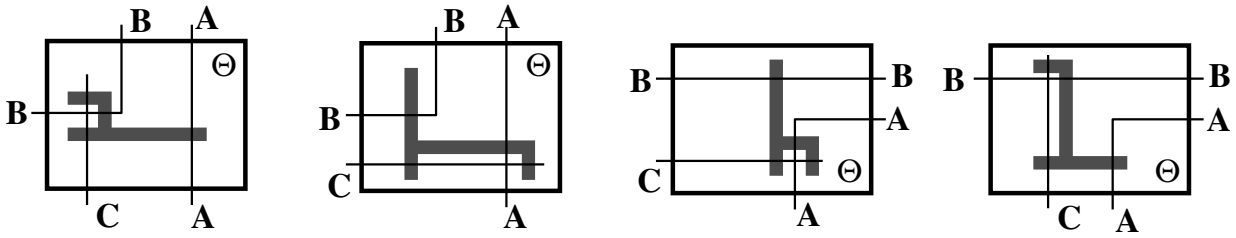
(c) reverse 3-sided layout

(d) A W-triangulation with marked corners.

Figure 3.2: Illustration of a 2-sided layout, 3-sided layout, and reverse 3-sided layout matching a W-triangulation.



(a) 2-sided representations for $F \in \{\{(A,C)\},\{(B,C)\},\emptyset\}$.



(b) 3-sided and reverse 3-sided representations for $F \in \{\{(B,C)\},\{(A,C)\}\}$.

Figure 3.3: $(int \cup F)$ representations of a triangle. Chair-shaped private regions are shaded in dark grey.

3.1.2 Private regions

Our proof starts by constructing a 1-string B_2-VPG representation for maximal planar graphs without separating triangles. The construction is then extended to all maximal planar graphs by merging representations of subgraphs obtained by splitting at separating triangles. To permit the merge, we apply the technique used in [31] (and also used independently in [42]): With every triangular face, create a region that intersects the curves of vertices of the face in a predefined way and does not intersect anything else, specifically not any such region of another face. Following the notation of [42], we call this a "private region" (but we use a different shape).

Definition 3.9 (Chair-shape)**.** *A* chair-shaped area *is a region bounded by a 10-sided orthogonal polygon with CW (clockwise) or CCW (counter-clockwise) sequence of interior angles* 90°, 90°, 270°, 270°, 90°, 90°, 90°, 90°, 270°, 90°. *See also Figure 3.4.*

Definition 3.10 (Private region)**.** *Let G be a planar graph with a partial* 1*-string B_2-VPG representation R and let f be a facial triangle in G. A* private region *of f is a chair-shaped area Φ inside R such that:*

1. *Φ is intersected by no curves except for the ones representing vertices on f.*
2. *All the intersections of R are located outside of Φ.*
3. *For a suitable labeling of the vertices of f as $\{a, b, c\}$, Φ is intersected by two segments of $\mathbf{a}$ and one segment of $\mathbf{b}$ and $\mathbf{c}$. The intersections between these segments and Φ occur at the edges of Φ as depicted in Figure 3.4.*

Figure 3.3 shows private regions for face $\{A, B, C\}$ for all choices of layout type and edge-set F.

3.1.3 The tangling technique

Our constructions will frequently use the following "tangling technique". Consider a set of k vertical downward rays $\mathbf{s_1}, \mathbf{s_2}, \mathbf{s_3}, \ldots, \mathbf{s_k}$ placed beside each other in left to right order.

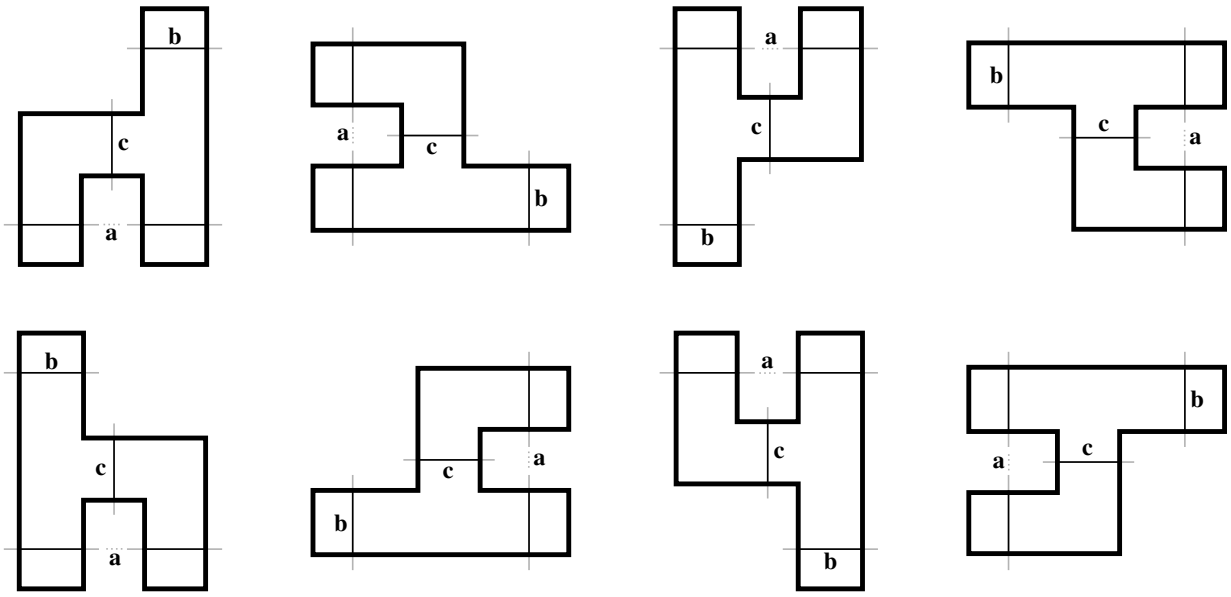


Figure 3.4: The chair-shaped private region of a triangle a, b, c with possible rotations and flips. Note that labels of a, b, c can be arbitrarily permuted—the curve intersecting the "base" of the chair does not need to be named **a**.

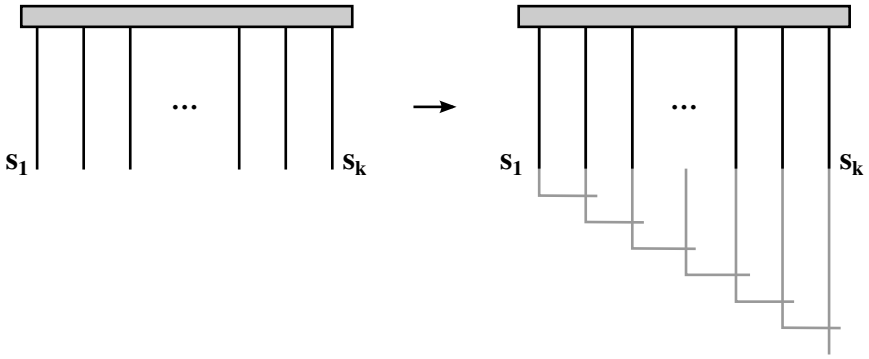


Figure 3.5: Bottom-tangling rightwards from $\mathbf{s_1}$ to $\mathbf{s_k}$ rightwards.

The operation of *bottom-tangling from* $\mathbf{s_1}$ *to* $\mathbf{s_k}$ *rightwards* stands for the following (see also Figure 3.5):

1. For $1 < i \leq k$, stretch $\mathbf{s_i}$ downwards so that it ends below $\mathbf{s_{i-1}}$.
2. For $1 \leq i < k$, bend $\mathbf{s_i}$ rightwards and stretch it so that it crosses $\mathbf{s_{i+1}}$, but so that it does not cross $\mathbf{s_{i+2}}$.

This creates intersections for the path $\mathbf{s_1}, \mathbf{s_2}, \mathbf{s_3}, \ldots, \mathbf{s_k}$.

We similarly define right-tangling upwards, top-tangling leftwards and left-tangling downwards as rotation of bottom-tangling rightwards by 90°, 180° and 270° CCW. We define bottom-tangling leftwards as a horizontal flip (i.e., along the y-axis) of bottom-tangling rightwards, and right-tangling downwards, top-tangling rightwards and left-tangling upwards as 90°, 180° and 270° CCW rotations of bottom-tangling leftwards.

3.2 2-sided constructions for W-triangulations

We first show the following lemma, which is the key result for Theorem 3.2 (representations of 4-connected planar graphs), and will also be used as an ingredient for the proof of Theorem 3.1 (representations of arbitrary planar graphs).

Lemma 3.11. *Let G be a W-triangulation. Let A, B, C be any three corners with respect to which G satisfies the chord condition, and let F be a set of at most one outerface edge incident to C. Then G has an $(int \cup F)$ 1-string B_2-VPG representation with 2-sided layout with respect to A, B. Furthermore, this representation has a chair-shaped private region for every interior face of G.*

We prove Lemma 3.11 by induction on the number of vertices.

First, let us make an observation that will greatly help to reduce the number of cases in the induction step. Define G^{rev} to be the graph obtained from graph G by reversing the combinatorial embedding, but keeping the same outerface. This effectively switches corners A and B, and replaces special edge (C, c_2) by (C, b_{s-1}) and vice versa. If G satisfies the chord condition with respect to corners (A, B, C), then G^{rev} satisfies the chord condition with respect to corners (B, A, C). (With this new order, the corners are CCW on the outerface of G^{rev}, as required.)

Presume we have a 2-sided representation of G^{rev}. Then we can obtain a 2-sided representation of G by flipping the one of G^{rev} horizontally. Hence for all the following cases, we may (after possibly applying the above flipping operation) make a restriction on which edge the special edge is.

Now we begin the induction. In the base case, $n = 3$, so G is a triangle, and the three corners A, B, C must be the three vertices of this triangle. The desired $(int \cup F)$ representations for all possible choices of F are depicted in Figure 3.3a.

The induction step for $n \geq 4$ is divided into three cases.

Case 1: C has degree 2

Figure 3.6 illustrates this case. Since G is a triangulated disk with $n \geq 4$, (b_{s-1}, c_2) is an edge. Define $G' := G - \{C\}$ and $F' := \{(b_{s-1}, c_2)\}$. We claim that G' satisfies the chord condition for corners $A' := A, B' := B$ and a suitable choice of $C' \in \{b_{s-1}, c_2\}$, and argue this as follows.

- If $c_2 = A$, then observe that $b_{s-1} \neq B$ as $n \geq 4$ and $deg(C) = 2$. Set $C' := b_{s-1}$. The chord condition holds for G' as b_{s-1} cannot be incident to a chord by planarity and the chord condition for G.

- If c_2 is incident to a chord that ends on P_{BC} other than (b_{s-1}, c_2), then $b_{s-1} \neq B$ is implied. Set $C' := b_{s-1}$. The chord condition holds for G' as b_{s-1} cannot be incident to a chord by planarity and the chord condition for G.

- Otherwise, $c_2 \neq A$ and c_2 is not incident to a chord that ends in an interior vertex of P_{BC} other than b_{s-1}, so set $C' := c_2$; clearly the chord condition holds for G'.

Thus in either case, we can apply induction to G'.

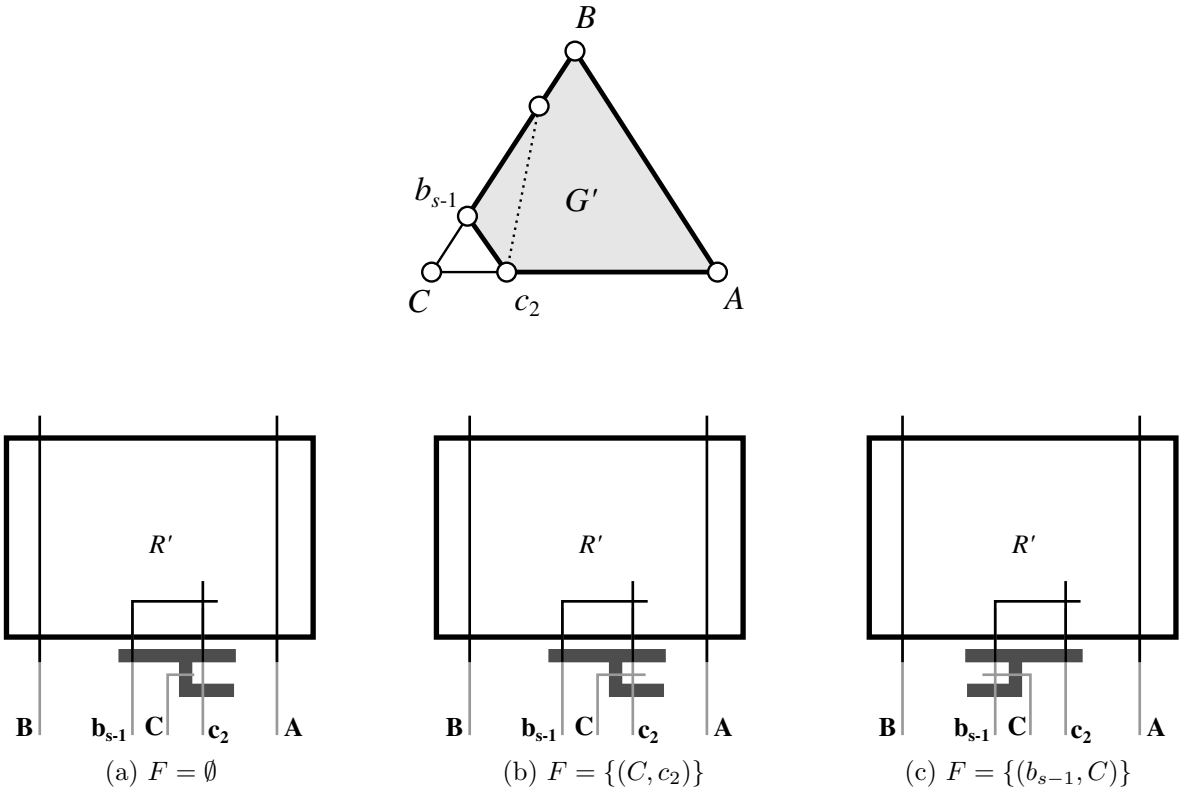
Refer to Figure 3.6. To create a 2-sided representation of G, we use a 2-sided $(int \cup F')$ representation R' of G' constructed with respect to the aforementioned corners. We introduce a new vertical curve $\mathbf{C}$ placed between $\mathbf{b_{s-1}}$ and $\mathbf{c_2}$ below R'. Add a bend at the upper end of $\mathbf{C}$ and extend it leftwards or rightwards. If the special edge e exists, then extend $\mathbf{C}$ until it hits the curve of the other endpoint of e; else extend it only far enough to allow for the creation of the private region.

With the exception of triangle $\{C, b_{s-1}, c_2\}$, all edges of G are interior/exterior in G if and only if they are interior/exterior in G', and hence represented by intersections as needed. Edge (c_2, b_{s-1}) is represented in R' by choice of F', and edges (b_{s-1}, C) and (C, c_2) are represented as needed. So, this is indeed a 2-sided $(int \cup F)$-representation.

Case 2: G has a chord incident to C

We may (after applying the reversal trick) assume that the special edge, if it exists, is (C, b_{s-1}). Refer to Figure 3.7.

By the chord condition, the chord incident to C has the form (C, a_i) for some $1 < i < r$. The graph G can be split along the chord (C, a_i) into two graphs G_1 and G_2. Both G_1 and G_2 are bounded by simple cycles, hence they are triangulated disks. No edges were added, so neither G_1 nor G_2 contains a separating triangle. So both of them are W-triangulations.



(a) $F = \emptyset$ (b) $F = \{(C, c_2)\}$ (c) $F = \{(b_{s-1}, C)\}$

Figure 3.6: Case 1: 2-sided construction if C has degree 2.

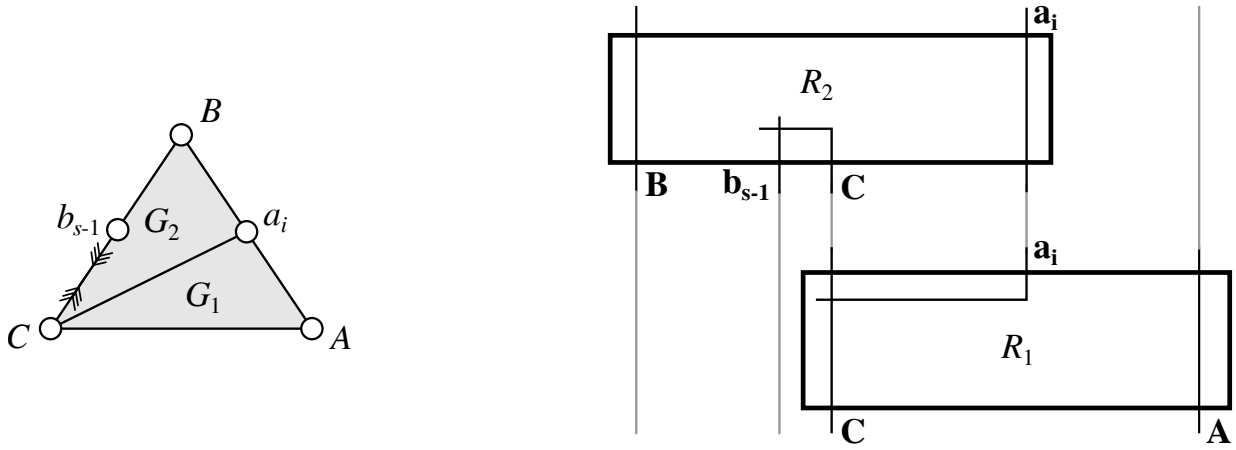


Figure 3.7: Case 2(a): Constructing an $(int \cup (C, b_{s-1}))$ representation when C is incident to a chord in 2-side layout. The special edge is marked with hatches.

We select (C, A, a_i) as corners for G_1 and (a_i, B, C) as corners for G_2 and can easily verify that G_1 and G_2 satisfy the chord condition with respect to those corners:

- G_1 has no chords on P_{Aa_i} or P_{CA} as they would violate the chord condition in G. There is no chord on P_{a_iC} as it is a single edge.
- G_2 has no chords on P_{a_iB} or P_{BC} as they would violate the chord condition in G. There is no chord on P_{a_iC} as it is a single edge.

Inductively construct a 2-sided $(int \cup (C, a_i))$ representation R_1 of G_1 and a 2-sided $(int \cup F)$ representation R_2 of G_2, both with the aforementioned corners. Note that $\mathbf{C}^{R_2}$ and $\mathbf{a_i}^{R_2}$ are consecutive on the bottom side of R_2 with $\mathbf{C}^{R_2}$ to the left of $\mathbf{a_i}^{R_2}$.

Rotate R_1 by 180°, and translate it so that it is below R_2 with $\mathbf{a_i}^{R_1}$ in the same column as $\mathbf{a_i}^{R_2}$. Stretch R_1 and R_2 horizontally as needed until $\mathbf{C}^{R_1}$ is in the same column as $\mathbf{C}^{R_2}$. Then $\mathbf{a_i}^R$ and $\mathbf{C}^R$ for $R \in \{R_1, R_2\}$ can each be unified without adding bends by adding vertical segments. The curves of outerface vertices of G then cross (after suitable lengthening) the bounding box in the required order.

Every interior face f of G is contained in G_1 or G_2 and hence has a private region in R_1 or R_2. As our construction does not make any changes inside the bounding boxes of R_1

and R_2, the private region of f is contained in R as well.

All edges are represented since F was represented in R_2, (C, a_i) was represented in R_1 and all other edges are interior/exterior in G if and only if they were interior/exterior in G_1 or G_2 and hence represented as needed.

Case 3: G has no chords incident to C and deg(C) ≥ 3

We may (after applying the reversal trick) assume that the special edge, if it exists, is (C, c_2).

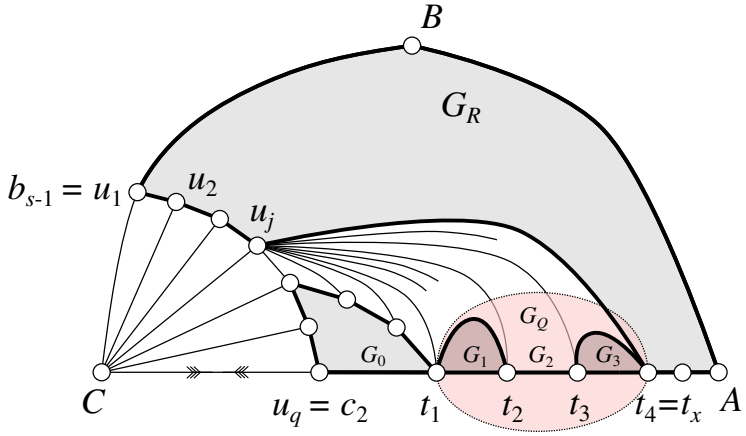
In this case we split G in a more complicated fashion illustrated in Figure 3.8. Let $u_1, \ldots, u_q$ be the neighbours of vertex C in clockwise order, starting with $b_{s-1} = u_1$ and ending with $c_2 = u_q$. We know that $q = deg(C) \geq 3$ and that $u_2, \ldots, u_{q-1}$ are not on the outerface, since C is not incident to a chord. Let u_j be a neighbour of C that has at least one neighbour other than C on P_{CA}, and among all those, choose j to be minimal. Such a j exists because G is a triangulated disk and therefore u_{q-1} is adjacent to both C and u_q.

We distinguish two sub-cases.

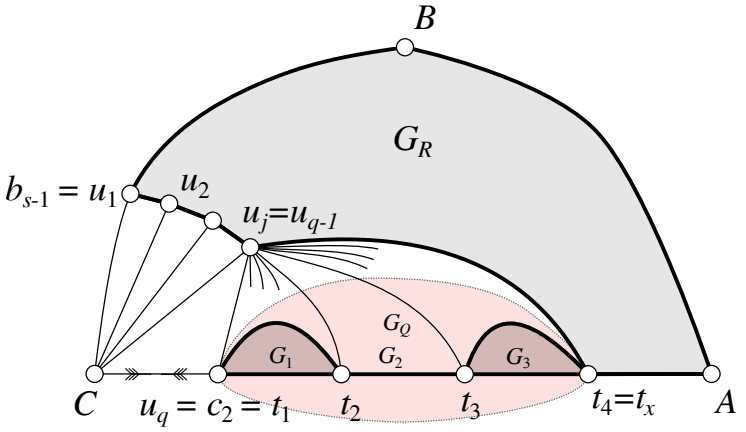
Case 3(a): $j \neq 1$. Denote the neighbours of u_j on P_{c_2A} by $t_1, \ldots, t_x$ in the order in which they appear on P_{c_2A}. Separate G into subgraphs as follows (see also Figure 3.8):

- The *right* graph G_R is bounded by $(A, \overset{P_{AB}}{\ldots}, B, \overset{P_{Bu_1}}{\ldots}, u_1, u_2, \ldots, u_j, t_x, \overset{P_{t_xA}}{\ldots}, A)$.
- Let G_B be the graph bounded by $(u_j, t_1, \overset{P_{t_1t_x}}{\ldots}, t_x, u_j)$. We are chiefly interested in its subgraph $G_Q := G_B - u_j$. If $t_1 = t_x$, then G_Q consists of just vertex t_1.
- Let G_L be the graph bounded by $(C, \overset{P_{Ct_1}}{\ldots}, t_1, u_j, C)$. We are chiefly interested in its subgraph $G_0 := G_L - \{u_j, C\}$. It may happen that G_0 consists of just c_2; this will be considered below.

The idea is to obtain representations of these subgraphs and then to combine them suitably. The following claim will be helpful to argue that the chord condition holds in for the subgraphs with respect to the chosen corners.



(a) $j < q - 1$; G_0 is non-trivial



(b) $j = q - 1$; $G_0 = \{c_2\}$

Figure 3.8: Case 3(a): Splitting the graph when $deg(C) \geq 3$, no chord is incident to C, and $j > 1$.

Claim 3.12. *Let G be a W-triangulation and v be a vertex of degree $k \geq 3$ on the outerface of G with neighbours $w_1, w_2, w_3, \ldots, w_k$ in clockwise order where w_1 and w_k are on the outerface of G as well. The subgraph induced by $w_1, w_2, w_3, \ldots, w_k$ contains only edges of the form (w_i, w_{i+1}).*

Proof. Assume that the graph contains an edge (w_i, w_j) and there is a w_x with $i < x < j$. Then w_i, w_j and v form a separating triangle with x in its interior, so G is not a W-triangulation. □

We first explain how to obtain the representation R_R used for G_R. Clearly G_R is a W-triangulation, since $u_2, \ldots, u_j$ are interior vertices of G, and hence the outerface of G_R is a simple cycle. Set $A_R := A$ and $B_R := B$. If $B \neq u_1$ then set $C_R := u_1$ and observe that G_R satisfies the chord condition with respect to these corners:

- G_R does not have any chords with both ends on $P_{A_R B_R} = P_{AB}$, $P_{B_R u_1} \subseteq P_{BC}$, or $P_{t_x A_R} \subseteq P_{CA}$ since G satisfies the chord condition.
- If there were any chords between a vertex in $u_1, \ldots, u_j$ and a vertex on $P_{C_R A_R}$, then by $C_R = u_1$ the chord would either connect two neighbours of C (hence giving a separating triangle of G; see also Claim 3.12), or connect some u_i for $i < j$ to P_{CA} (contradicting the minimality of j), or connect u_j to some other vertex on $P_{t_x A}$ (contradicting that t_x is the last neighbour of u_j on P_{CA}). Hence no such chord can exist either.

If $B = u_1$, then set $C_R := u_2$ (which exists by $q \geq 3$) and similarly verify that it satisfies the chord condition as $P_{B_R C_R}$ is the edge (B, u_2). Since $C_R \in \{u_1, u_2\}$ in both cases, we can apply induction on G_R and obtain a 2-sided $(int \cup (u_1, u_2))$ representation R_R with respect to the aforementioned corners.

Next we obtain a representation for the graph G_0, which is bounded by $u_{j+1}, \ldots, u_q, P_{c_2 t_1}$ and the neighbours of u_j between t_1 and u_{j+1} in CW order around u_j. We distinguish two cases:

(1) $j = q - 1$, and hence $t_1 = u_q = c_2$ and G_0 consists of only c_2 (see Figure 3.8b). In this case, the representation of R_0 consists of a single vertical line segment $\mathbf{c_2}$.

(2) $j < q - 1$, so G_0 contains at least three vertices u_{q-1}, u_q and t_1. Then G_0 is a W-triangulation since C is not incident to a chord and by the choice of t_1. Also, it satisfies the chord condition with respect to corners $A_0 := c_2, B_0 := t_1$ and $C_0 := u_{j+1}$ since the three paths on its outerface are sub-paths of P_{CA} or contained in the neighbourhood of C or u_j. In this case, construct a 2-sided $(int \cup (u_{j+1}, u_{j+2}))$ representation R_0 of G_0 with respect to these corners inductively.

Finally, we create a representation R_Q of G_Q. If G_Q is a single vertex or a single edge, then simply use vertical segments for the curves of its vertices (recall that there is no special edge in G_Q, so none of its outer edges need to be represented by crossings). Otherwise, we can show:

Claim 3.13. *G_Q has a 2-sided $(int \cup \emptyset)$ 1-string B_2-VPG representation with respect to corners t_1 and t_x.*

Proof. G_Q is not necessarily 2-connected, so we cannot apply induction directly. Instead we break it into $x - 1$ graphs $G_1, \ldots, G_{x-1}$, where for $i = 1, \ldots, x - 1$ graph G_i is bounded by $P_{t_i t_{i+1}}$ as well as the neighbours of u_j between t_i and t_{i+1} in CCW order (see Figure 3.9(a)). Note that G_i is either a single edge, or it is bounded by a simple cycle since u_j has no neighbours on P_{CA} between t_i and t_{i+1}.

First, obtain a representation R_i of G_i as follows. If G_i is a single edge (t_i, t_{i+1}), then let R_i consists of two vertical segments $\mathbf{t_i}$ and $\mathbf{t_{i+1}}$. Otherwise, define three corners of G_i to be $B_i := t_i$, $A_i := t_{i+1}$, and C_i an arbitrary third vertex on $P_{t_i t_{i+1}} \subseteq P_{CA}$. This vertex C_i exists since the outerface of G_i is a simple cycle and (t_i, t_{i+1}, u_j) is not a separating triangle. Observe that G_i satisfies the chord condition since all paths on the outerface of G_i are either part of P_{CA} or in the neighbourhood of u_j. Hence by induction there exists a 2-sided $(int \cup \emptyset)$ representation R_i of G_i with respect to the corners of G_i.

Since each representation R_i has at its leftmost end a vertical segment $\mathbf{t_i}$ and at its rightmost end a vertical segment $\mathbf{t_{i+1}}$, we can combine all these representations by aligning

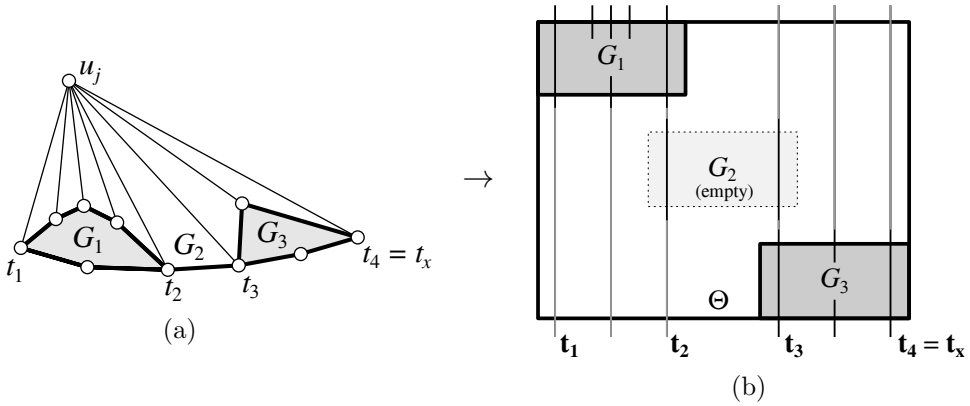


Figure 3.9: (a) Graph G_B. The boundary of G_Q is shown bold. (b) Merging 2-sided ($int \cup \emptyset$) representations of $G_i, 1 \leq i \leq 3$, into a 2-sided ($int \cup \emptyset$) representation of G_Q.

$\mathbf{t_i}^{R_i}$ and $\mathbf{t_i}^{R_{i+1}}$ horizontally and filling in the missing segment. See also Figure 3.9(b). One easily verifies that the result is a 2-sided ($int \cup \emptyset$) representation of G_Q. □

We now explain how to combine these three representations R_R, R_Q and R_0; see also Figure 3.10. Translate R_Q so that it is below R_R with $\mathbf{t_x}^{R_R}$ and $\mathbf{t_x}^{R_Q}$ in the same column; then connect these two curves with a vertical segment. Rotate R_0 by 180° and translate it so that it is below R_R and to the left and above R_Q, and $\mathbf{t_1}^{R_o}$ and $\mathbf{t_1}^{R_Q}$ are in the same column; then connect these two curves with a vertical segment. Notice that the vertical segments of $\mathbf{u_2}^{R_R}, \ldots, \mathbf{u_j}^{R_R}$ are at the bottom left of R_R. Horizontally stretch R_0 and/or R_R so that $\mathbf{u_2}^{R_R}, \ldots, \mathbf{u_j}^{R_R}$ are to the left of the vertical segment of $\mathbf{u_{j+1}}^{R_o}$, but to the right (if $j < q-1$) of the vertical segment of $\mathbf{u_{j+2}}^{R_o}$. There are such segments by $j > 1$.

Introduce a new horizontal segment $\mathbf{C}$ and place it so that it intersects curves $\mathbf{u_q}, \ldots, \mathbf{u_{j+2}}$, $\mathbf{u_2}, \ldots, \mathbf{u_j}, \mathbf{u_{j+1}}$ (after lengthening them, if needed), but omit the intersection with u_q if the special edge $(C, u_q = c_2)$ does not exist (see Figures 3.10 (b)). Attach a downward vertical segment to $\mathbf{C}$ at the left end. If $j < q-1$, then top-tangle $\mathbf{u_q}, \ldots, \mathbf{u_{j+2}}$ rightwards. (Recall from Section 3.1.3 that this creates intersections among all these curves.) Bottom-tangle $\mathbf{u_2}, \ldots, \mathbf{u_j}$ rightwards. The construction hence creates intersections for all edges in the path $u_1, \ldots, u_q$, except for (u_{j+2}, u_{j+1}) (which was represented in R_0), (u_2, u_1) (which was

represented in R_R), and (u_j, u_{j+1}) (which we represent below).

Bend and stretch $\mathbf{u_j}^{R_R}$ rightwards so that it crosses the curves of all its neighbours in $G_0 \cup G_Q$; this includes u_{j+1}. Finally, consider the path between the neighbours of u_j CCW from u_{j+1} to t_x. Top-tangle curves of these vertices rightwards, but omit the intersection if the edge is on the outerface (see e.g. (t_2, t_3) in Figure 3.10).

One verifies that all the edges are represented and the curves intersect the bounding boxes as desired. The constructed representations contain private regions for all interior faces of G_R, G_Q and G_0 by induction. The remaining faces are of the form $(C, u_i, u_{i+1}), 1 \leq i < q$, and (u_j, w_k, w_{k+1}) where w_k and w_{k+1} are two consecutive neighbours of u_j on the outerface of G_0 or G_Q. Private regions for those faces are shown in Figure 3.10.

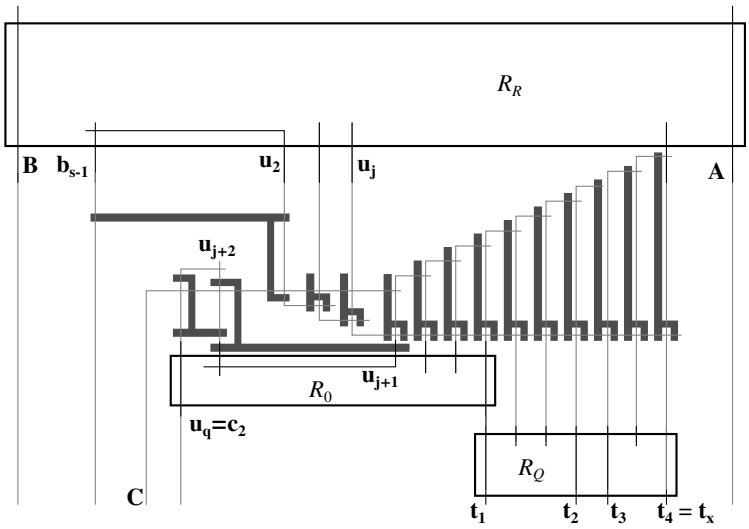
Case 3(b): $j = 1$, i.e., there exists a chord (b_{s-1}, c_i) for some $c_i \in P_{CA}$. In this case we cannot use the above construction directly since it bends $\mathbf{u_j} = \mathbf{u_1} = \mathbf{b_{s-1}}$ horizontally rightwards to create intersections, but then $\mathbf{u_j}$ no longer extends vertically downwards as required for $\mathbf{b_{s-1}}$. Instead we use a different construction, illustrated in Figure 3.11.

Edge (b_{s-1}, c_i) is a chord from P_{BC} to P_{CA}. Let (b_k, c_ℓ) be a chord from P_{BC} to P_{CA} that maximizes $k - \ell$, i.e., is furthest from C (our construction in this case actually works for any chord from P_{BC} to P_{CA}—it is not necessary that $k = s - 1$). Note that possibly $\ell = t$ (i.e., the chord is incident to A) or $k = 1$ (i.e., the chord is incident to B), but not both by the chord condition. We assume here that $\ell < t$, the other case is symmetric.

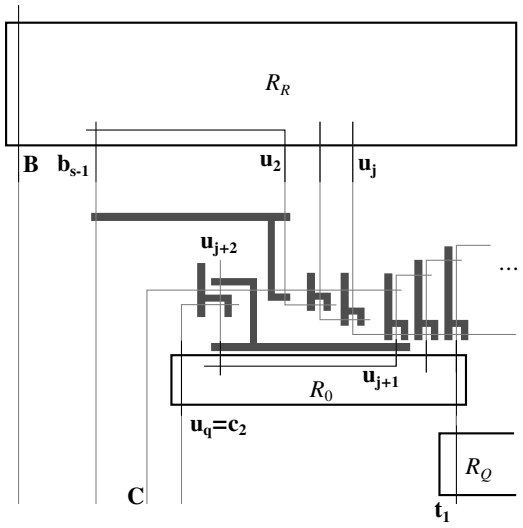
In order to construct a 2-sided $(int \cup F)$ representation of G, split the graph along (b_k, c_ℓ) into two W-triangulations G_1 (which includes C and the special edge, if any) and G_2 (which includes A). Set (A, B, c_ℓ) as corners for G_2 (these are three distinct vertices by $c_\ell \neq A$) and set (c_ℓ, b_k, C) as corners for G_1 and verify the chord condition:

- G_1 has no chords on either $P_{Cc_\ell} \subseteq P_{CA}$ or $P_{b_kC} \subseteq P_{BC}$ as they would contradict the chord condition in G. The third side is a single edge (b_k, c_ℓ) and so it does not have any chords either.

- G_2 has no chords on either $P_{c_\ell A} \subseteq P_{CA}$ or P_{AB} as they would violate the chord condition in G. It does not have any chords on the path P_{Bc_ℓ} due to the selection of the chord (b_k, c_ℓ) and by the chord condition in G.



(a) $F = \{(C, c_2)\}$



(b) $F = \emptyset$

Figure 3.10: Combining subgraphs in Case 3(a), 2-sided construction. The construction matches the graph depicted in Figure 3.8a.

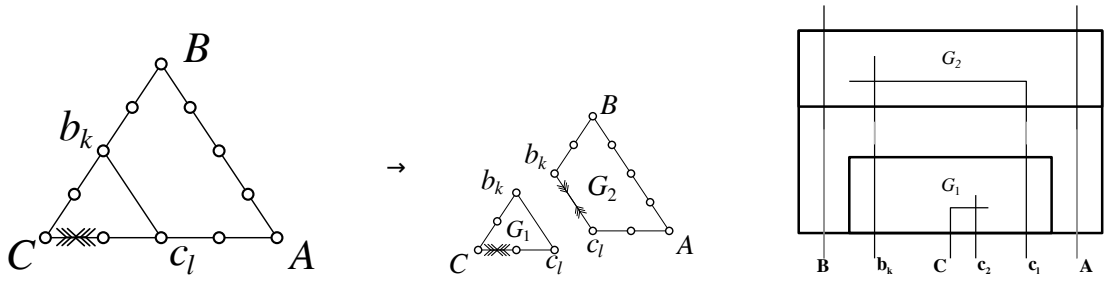


Figure 3.11: Case 3(b): Construction of a 2-sided $(int \cup (C, c_2))$ representation of G with a chord (b_k, c_ℓ).

Thus, by induction, G_1 has a 2-sided $(int \cup F)$ representation R_1 and G_2 has a 2-sided $(int \cup (b_k, c_\ell))$ representation R_2 with respect to the aforementioned corners. Translate and horizontally stretch R_1 and/or R_2 so that $\mathbf{b_k^{R_1}}$ and $\mathbf{c_\ell^{R_1}}$ are aligned with $\mathbf{b_k^{R_2}}$ and $\mathbf{c_\ell^{R_2}}$, respectively, and connect each pair of curves with a vertical segment. Since $\mathbf{b_k^{R_1}}$ and $\mathbf{c_\ell^{R_1}}$ have no bends, this does not increase the number of bends on any curve and produces a 2-sided $(int \cup F)$ representation of G. All the faces in G have a private region inside one of the representations of G_1 or G_2.

This ends the description of the construction in all cases, and hence proves Lemma 3.11. We now show how Lemma 3.11 implies Theorem 3.2:

Proof of Theorem 3.2. Let G be a 4-connected planar graph. Assume first that G is triangulated, which means that it is a W-triangulation. Let (A, B, C) be the outerface vertices and start with an $(int \cup (B, C))$-representation of G (with respect to corners (A, B, C)) that exists by Lemma 3.11. The intersections of the other two outerface edges (A, C) and (A, B) can be created by tangling $\mathbf{B}, \mathbf{A}$ and $\mathbf{C}, \mathbf{A}$ suitably (see Figure 3.12).

Theorem 3.2 also stipulates that every curve used in a representation has at most one vertical segment. This is true for all curves added during the construction. Furthermore, we join two copies of a curve only by aligning and connecting their vertical ends, so all curves have at most one vertical segment.

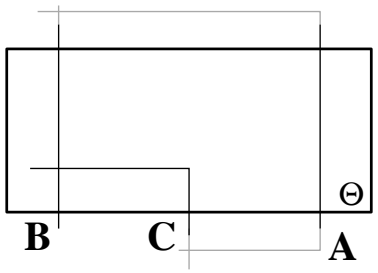


Figure 3.12: Completing a 2-sided $(int \cup (B, C))$ representation by adding intersections for (A, B) and (A, C).

This proves Theorem 3.2 for 4-connected triangulations. To handle an arbitrary 4-connected planar graph, *stellate* the graph, i.e., insert into each non-triangular face f a new vertex v and connect it to all vertices on f. By 4-connectivity this creates no separating triangle and the graph is triangulated afterwards. Finding a representation of the resulting graph and deleting the curves of all added vertices yields the result. □

3.3 3-sided constructions for W-triangulations

In the previous section, we proved the existence of B_2-VPG representations with 2-sided layout for 4-connected planar graphs. However, in order to show the existence of B_2-VPG representations for all planar graphs (Theorem 3.1), we will later also need B_2-VPG representations with 3-sided layouts. The proof of this is similar in spirit (distinguishing cases by degree and neighbourhood of C), but the constructions are different and we hence must redo all cases. Hence, we prove:

Lemma 3.14. *Let G be a W-triangulation and let A, B, C be any three corners with respect to which G satisfies the chord condition. For any $e \in \{(C, b_{s-1}), (C, c_2)\}$, G has an $(int \cup e)$ 1-string B_2-VPG representation with 3-sided layout and an $(int \cup e)$ 1-string B_2-VPG representation with reverse 3-sided layout. Both representations have a chair-shaped private region for every interior face.*

The proof of Lemma 3.14 will use induction on the number of vertices. To combine the representations of subgraphs, we sometimes need them to have a 2-sided layout, and hence we frequently use Lemma 3.11 proved in Section 3.2. Also, notice that for Lemma 3.14 the special edge *must* exist (this is needed in Case 1 to find private regions), while for Lemma 3.11, F is allowed to be empty.

We again reduce the number of cases in the proof of Lemma 3.14 by using the reversal trick. Define G^{rev} as in Section 3.2. Presume we have a 3-sided/reverse 3-sided representation of G^{rev}. We can obtain a 3-sided/reverse 3-sided representation of G by flipping the reverse 3-sided/3-sided representation of G^{rev} diagonally (i.e., along the line defined by $(x = y)$). Again, this effectively switches corners A and B (corner C remains the same), and replaces special edge (C, c_2) by (C, b_{s-1}) and vice versa. If G satisfies the chord condition with respect to corners (A, B, C), then G^{rev} satisfies the chord condition with respect to corners (B, A, C). Hence for all the following cases, we may again (after possibly applying the above flipping operation) make a restriction on which edge the special edge is. Alternatively, we only need to give the construction for the 3-sided, but not for the reverse 3-sided layout.

So let G and a special edge e be given, and set $F = \{e\}$. In the base case, $n = 3$, so G is a triangle, and the three corners A, B, C must be the three vertices of this triangle. The desired $(int \cup F)$ representations for all possible choices of F are depicted in Figure 3.3a. The induction step for $n \geq 4$ uses the same case distinctions as the proof of Lemma 3.11.

Case 1: C has degree 2

Since G is a triangulated disk with $n \geq 4$, (b_{s-1}, c_2) is an edge. Define G' as in Section 3.2 to be $G - \{C\}$ and recall that G' satisfies the chord condition for corners $A' := A, B' := B$ and a suitable choice of $C' \in \{b_{s-1}, c_2\}$. Thus, we can apply induction to G'.

To create a 3-sided representation of G, we use a 3-sided $(int \cup F')$ representation R' of G', where $F' = \{(b_{s-1}, c_2)\}$. Note that regardless of which vertex is C', we have $\mathbf{b_{s-1}}$ as the bottommost curve on the left and $\mathbf{c_2}$ as the leftmost curve on the bottom. Introduce a new horizontal segment representing C which intersects $\mathbf{c_2}$ if $F = \{(C, c_2)\}$, or a vertical segment which intersects $\mathbf{b_{s-1}}$ if $F = \{(C, b_{s-1})\}$.

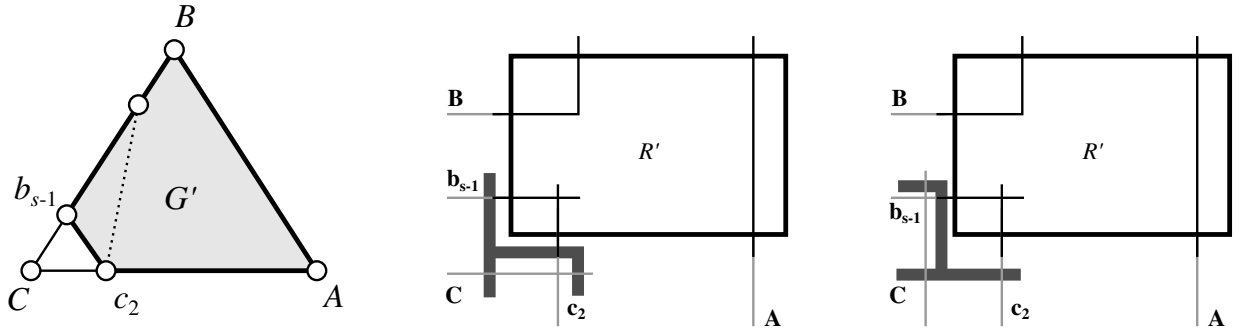


Figure 3.13: Case 1: 3-sided representation if C has degree 2.

After suitable lengthening, the curves intersect the bounding box in the required order. One can find the chair-shaped private region for the only new face $\{C, c_2, b_{s-1}\}$ as shown in Figure 3.13. Observe that no bends were added to the curves of R' and that C has no bends as required.

Since we have given the constructions for both possible special edges, we can obtain the reverse 3-sided representation by diagonally flipping a 3-sided representation of G^{rev}.

Case 2: G has a chord incident to C

Let (C, a_i) be a chord that minimizes i (i.e., is closest to A). Define W-triangulations G_1 and G_2 with corners (C, A, a_i) for G_1 and (a_i, B, C) for G_2 as in Section 3.2 (see also Figure 3.14), and recall that they satisfy the chord condition. So, we can apply induction to both G_1 and G_2, obtain representations R_1 and R_2 (with respect to the aforementioned corners) for them, and combine them suitably. We will do so for both possible choices of special edge, and hence need not give the constructions for reverse 3-sided layout due to the reversal trick.

Case 2(a): $F = \{(C, b_{s-1})\}$**.** Using Lemma 3.11, construct a 2-sided $(int \cup (C, a_i))$ representation R_1 of G_1 with respect to the aforementioned corners of G_1. Inductively, construct a 3-sided $(int \cup F)$ representation R_2 of G_2 with respect to the corners of G_2. Note that $\mathbf{C}^{R_2}$ and $\mathbf{a_i}^{R_2}$ are on the bottom side of R_2 with $\mathbf{C}^{R_2}$ to the left of $\mathbf{a_i}^{R_2}$.

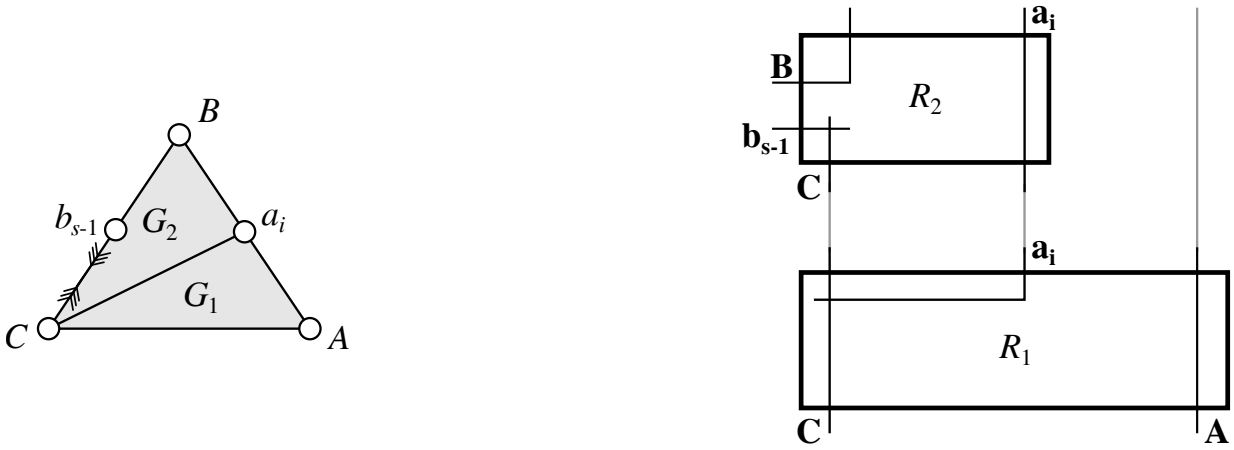


Figure 3.14: Case 2(a): Constructing a 3-sided $(int \cup (C, b_{s-1}))$ representation when C is incident to a chord.

Rotate R_1 by 180°. We can now merge R_1 and R_2 as described in Section 3.2 since all relevant curves end vertically in R_1 and R_2. The curves of outerface vertices of G then cross (after suitable lengthening) the bounding box in the required order. See also Figure 3.14.

Case 2(b): $F = \{(C, c_2)\}$**.** For the 3-sided construction, it does not seem possible to merge suitable representations of G_1 and G_2 directly, since the geometric restrictions imposed onto curves $\mathbf{A}, \mathbf{B}, \mathbf{C}, \mathbf{c_2}$ and $\mathbf{a_i}$ by the 3-sided layout cannot be satisfied using 3-sided and 2-sided representations of G_1 and G_2. We hence use an entirely different approach that splits the graph further; it resembles Case 1 in [7, Proof of Lemma 2] and is illustrated in Figures 3.15 and 3.16. Let $G_Q = G_1 - C$, and observe that it is bounded by P_{c_2A}, P_{A,a_i}, and the path formed by the neighbours $c_2 = u_1, u_2, \ldots, u_q = a_i$ of C in G_1 in CCW order. We must have $q \geq 2$, but possibly G_1 is a triangle $\{C, A, a_i\}$ and G_Q then degenerates into an edge. If G_Q contains at least three vertices, then $u_2, \ldots, u_{q-1}$ are interior since chord (C, a_i) was chosen closest to A, and so G_Q is a W-triangulation.

We divide the proof into two subcases, depending on whether $A \neq c_2$ or $A = c_2$.

Case 2(b)1: $A \neq c_2$**.** Select the corners of G_Q as $(A_Q := c_2, B_Q := A, C_Q := a_i = u_q)$, and observe that it satisfies the chord condition since the three corners are distinct and the three outerface paths are sub-paths of P_{CA} and P_{AB} or in the neighbourhood of C,

respectively. Apply Lemma 3.11 to construct a 2-sided $(int \cup (u_q, u_{q-1}))$ representation R_Q of G_Q with respect to the previously chosen corners of G_Q. Inductively, construct a 3-sided $(int \cup (C, a_i))$ representation R_2 of G_2 with respect to the previously chosen corners (a_i, B, C).

To combine R_Q with R_2, rotate R_Q by 180°. Appropriately stretch R_Q and translate it so that it is below R_2 with $\mathbf{a_i}^{R_Q}$ and $\mathbf{a_i}^{R_2}$ in the same column, and so that the vertical segment of each of the curves $\mathbf{u_{q-1}}, \ldots, \mathbf{u_1} = \mathbf{c_2}$ is to the left of the bounding box of R_2. Then $\mathbf{a_i}^{R_Q}$ and $\mathbf{a_i}^{R_2}$ can be unified without adding bends by adding a vertical segment. Curves $\mathbf{u_{q-1}}, \ldots, \mathbf{u_1} = \mathbf{c_2}$ in the rotated R_Q can be appropriately stretched upwards, intersected by $\mathbf{C}^{R_2}$ after stretching it leftwards, and then top-tangled leftwards. All the curves of outerface vertices of G then cross (after suitable lengthening) a bounding box in the required order.

All faces in G that are not interior to G_Q or G_2 are bounded by (C, u_k, u_{k+1}), $1 \leq k < q$. The chair-shaped private regions for such faces can be found as shown in Figure 3.15.

Case 2(b)2: $A = c_2$**.** In this case the previous construction cannot be applied since the corners for G_Q would not be distinct. We give an entirely different construction.

If G_Q has at least 3 vertices, then $q \geq 3$ since otherwise by $A = c_2 = u_1$ edge (A, u_q) would be a chord on P_{AB}. Choose as corners for G_Q the vertices $A_Q := A, B_Q := a_i = u_q$ and $C_Q := u_{q-1}$ and observe that the chord condition holds since all three paths on the outerface belong to P_{AB} or are in the neighbourhood of C. By Lemma 3.11, G_Q has a 2-sided $(int \cup (u_q, u_{q-1}))$ representation R_Q with the respective corners and private region for every interior face of G_Q. If G_Q has at most 2 vertices, then G_Q consists of edge (A, a_2) only, and we use as representation R_2 two parallel vertical segments $\mathbf{a_2}$ and $\mathbf{A}$.

We combine R_Q with a representation R_2 of G_2 that is *different* from the one used in the previous cases; in particular we rotate corners. Construct a reverse 3-sided layout R_2 of G_2 with respect to corners $C_2 := a_i$, $A_2 := B$ and $B_2 := C$. Rotate R_2 by 180°, and translate it so that it is situated below R_Q with $\mathbf{a_i}^{R_Q}$ and $\mathbf{a_i}^{R_2}$ in the same column. Then, extend $\mathbf{C}^{R_2}$ until it crosses $\mathbf{u_{q-1}}^{R_Q}, \ldots, \mathbf{u_1}^{R_Q}$ (after suitable lengthening), and then bottom-tangle $\mathbf{u_{q-1}}^{R_Q}, \ldots, \mathbf{u_1}^{R_Q}$ rightwards. This creates intersections for all edges in path $u_q, u_{q-1}, \ldots, u_1$, except for (u_q, u_{q-1}), which is either on the outerface (if $q = 2$) or had an intersection in

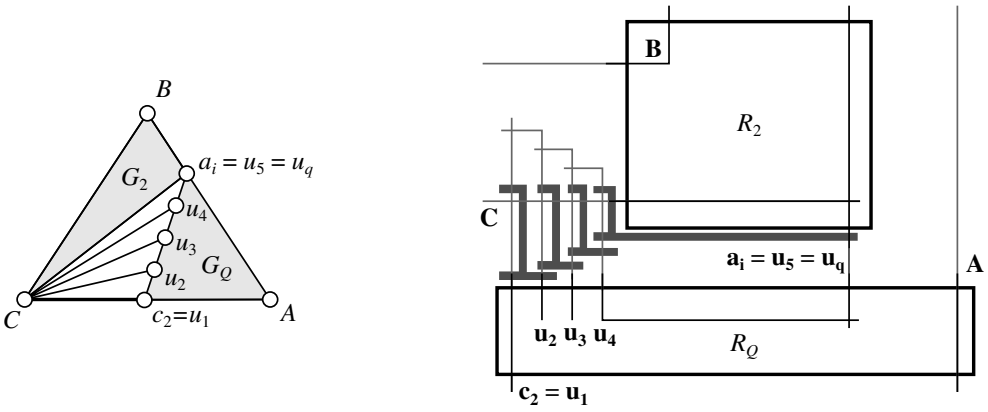


Figure 3.15: Case 2(b)1: C is incident to a chord, $F = \{(C, c_2)\}$, and $c_2 \neq A$.

R_Q. One easily verifies that the result is a 3-sided layout, and private regions can be found for the new interior faces as shown in Figure 3.16.

Case 3: G has no chords incident to C and deg(C) ≥ 3

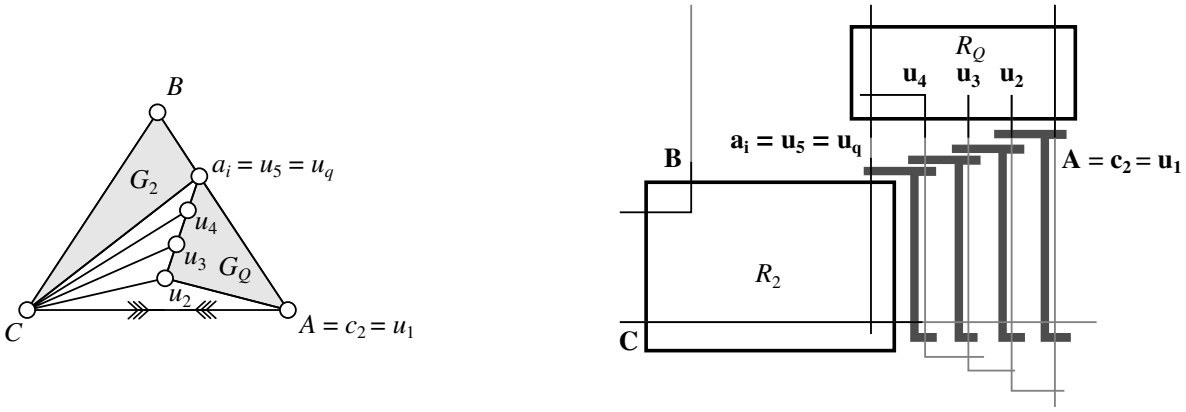
We will give explicit constructions for 3-sided and reverse 3-sided layout, and may hence (after applying the reversal trick) assume that the special edge is (C, c_2).

As in Section 3.2, let $u_1, \ldots, u_q$ be the neighbours of C and let j be minimal such that u_j has another neighbour on P_{AC}. We again distinguish two sub-cases.

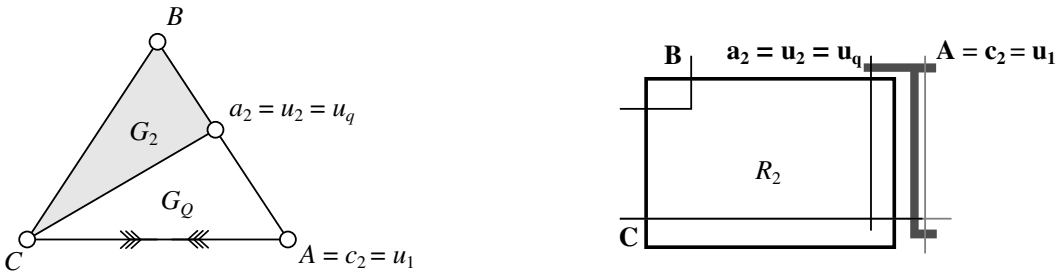
Case 3(a): $j \neq 1$. As in Section 3.2, define $t_1, \ldots, t_x$, G_R, G_B, G_Q, G_L and G_0. See also Figure 3.8 on page 39. Recall that G_R satisfies all conditions with respect to corners $A_R := A$, $B_R := B$ and $C_R \in \{u_1, u_2\}$. Apply induction on G_R and obtain an $(int \cup (u_1, u_2))$ representation R_R with respect to the corners of G_R. We use as layout for R_R the type that we want for G, i.e., use a 3-sided/reverse 3-sided layout if we want G to have a 3-sided/reverse 3-sided representation.

For G_0 and G_Q, we use exactly the same representations R_0 and R_Q as in Section 3.2.

Combine now these three representations R_R, R_Q and R_0 as described in Section 3.2, Case 3(a); this can be done since the relevant curves $\mathbf{u_2^{R_R}}, \ldots, \mathbf{u_t^{R_R}}$ all end vertically in R_R.



(a) (A, a_i, C) is not a face



(b) (A, a_i, C) is a face

Figure 3.16: Case 2(b)2: Construction when C is incident to a chord, $c_2 = A$ and $F = \{(C, c_2)\}$.

See also Figure 3.17. The only change occurs at curve **C**; in Section 3.2 this received a bend and a downward segment, but here we omit this bend and segment and let **C** end horizontally as desired.

One easily verifies that the curves intersect the bounding boxes as desired. The constructed representations contain private regions for all interior faces of G_R, G_Q and G_0 by induction. The remaining faces are of the form $(C, u_i, u_{i+1}), 1 \leq i < q$, and (u_j, w_k, w_{k+1}) where w_k and w_{k+1} are two consecutive neighbours of u_j on the outerface of G_0 or G_Q. Private regions for those faces are shown in Figure 3.17.

Case 3(b): $j = 1$, i.e., there exists a chord (b_{s-1}, c_i). In this case we cannot use the above construction directly since we need to bend $\mathbf{u_j = u_1 = b_{s-1}}$ horizontally rightwards to create intersections, but then it no longer extends vertically downwards as required for $\mathbf{b_{s-1}}$. The simple construction described in Section 3.2, Case 3(b) does not apply either. However, if we use a different vertex as u_j (and argue carefully that the chord condition holds), then the same construction works.

Refer to Figure 3.18. Recall that $u_1, \ldots, u_q$ are the neighbours of corner C in CW order starting with b_{s-1} and ending with c_2. We know that $q \geq 3$ and $u_2, \ldots, u_{q-1}$ are not on the outerface. Now define j' as follows: Let $u_{j'}$, $j' > 1$ be a neighbour of C that has at least one neighbour on P_{CA} other than C, and choose $u_{j'}$ so that j' is minimal while satisfying $j' > 1$. Such a j' exists since u_{q-1} has another neighbour on P_{CA}, and by $q \geq 3$ we have $q - 1 > 1$. Now, separate G as in the previous case, except use j' in place of j. Thus, define $t_1, \ldots, t_x$ to be the neighbours of $u_{j'}$ on P_{c_2A}, in order, and separate G into three graphs as follows:

- The *right* graph G_R is bounded by $(A, \overset{P_{AB}}{\ldots}, B, \overset{P_{Bu_1}}{\ldots}, u_1, u_2, \ldots, u_{j'}, t_x, \overset{P_{t_xA}}{\ldots}, A)$.
- Let G_B be the graph bounded by $(u_{j'}, t_1, \overset{P_{t_1t_x}}{\ldots}, t_x, u_{j'})$. Define $G_Q := G_B - u_{j'}$.
- Let G_L be the graph bounded by $(C, \overset{P_{Ct_1}}{\ldots}, t_1, u_{j'}, C)$. Define $G_0 := G_L - \{u_{j'}, C\}$.

Observe that the boundaries of all the graphs are simple cycles, and thus they are W-triangulations. Select $(A_R := A, B_R := B, C_R := u_2)$ to be the corners of G_R and argue the chord condition as follows:

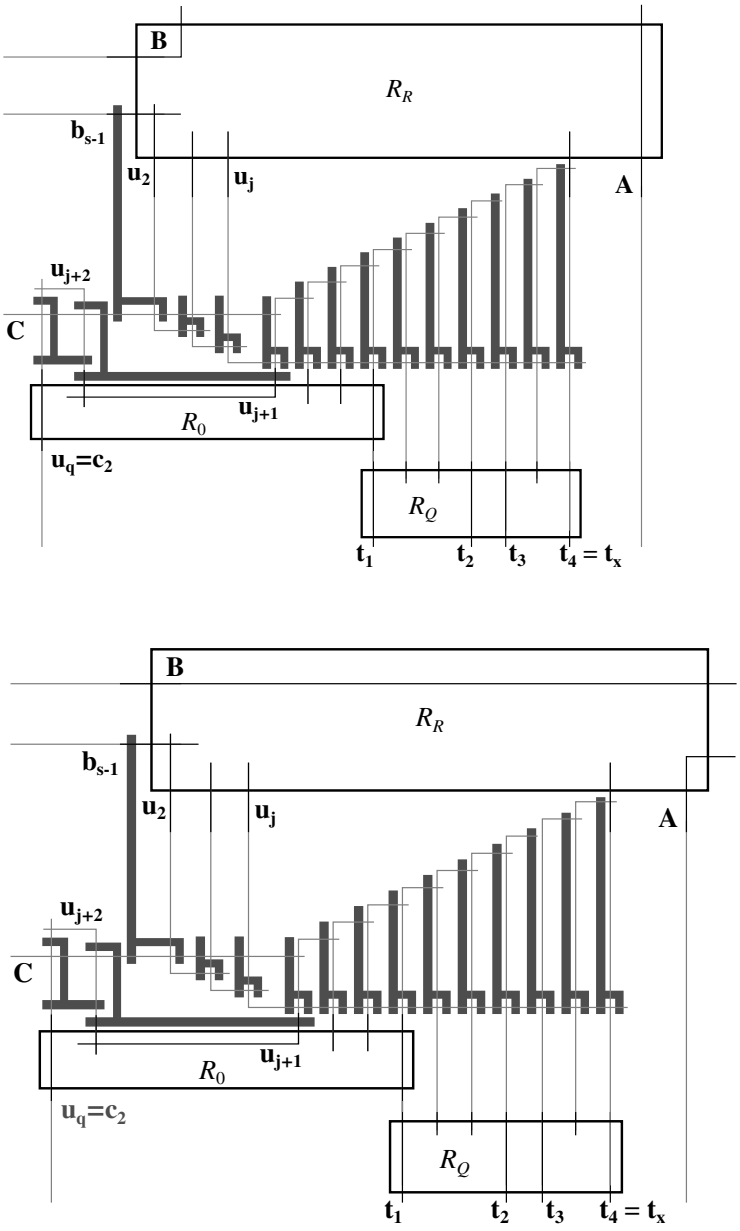


Figure 3.17: Case 3(a): 3-sided and reverse 3-sided representation when $deg(C) \geq 3$, there is no chord incident to C, $F = \{(C, c_2)\}$, and $j > 1$. The construction matches the graph depicted in Figure 3.8a.

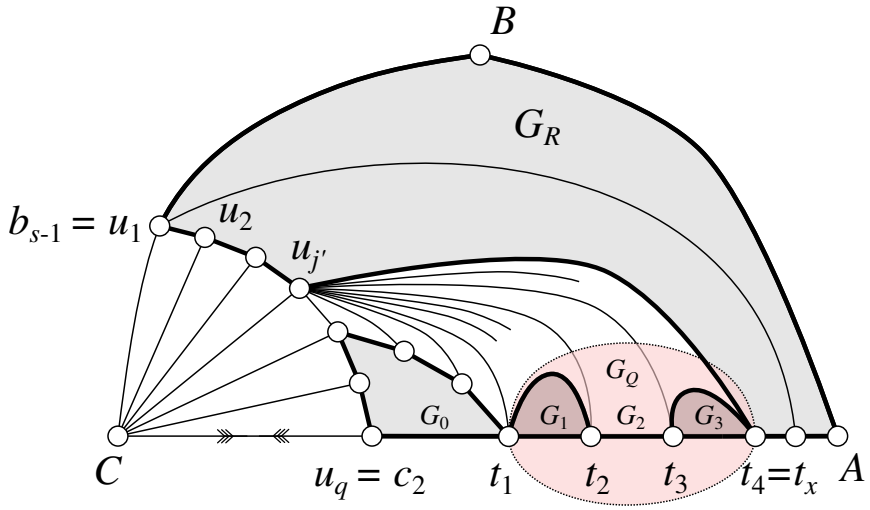


Figure 3.18: Case 3(b): Splitting the graph when $deg(C) \geq 3$, no chord is incident to C, and $j = 1$.

- G_R does not have any chords on $P_{C_R A_R}$ as such chords would either contradict the minimality of j', or form a separating triangle with C (see Claim 3.12) and violate the chord condition in G.
- G_R does not have any chords on $P_{A_R B_R} = P_{AB}$.
- G_R does not have any chords on $P_{Bb_{s-1}}$ as it is a sub-path of P_{BC} and they would violate the chord condition in G. It also does not have any chords in the form $(C_R = u_2, b_\ell), 1 \leq \ell < s-1$ as they would have to intersect the chord (b_{s-1}, c_i), violating the planarity of G. Hence, G_R does not have any chords on $P_{C_R A_R}$.
- Notice in particular that the chord (u_1, c_i) of G_R is *not* a violation of the chord condition since we chose u_2 as a corner.

Hence, we can obtain a representation R_R of G_R with 3-sided or reverse 3-sided layout and special edge $(u_1 = b_{s-1}, u_2)$. For graphs G_Q and G_0 the corners are chosen, the chord condition is verified, and the representations are obtained exactly as in Case 3(a). Since the special edge of G_R is (u_1, u_2) as before, curves $\mathbf{u_1}^{R_R}$ and $\mathbf{u_2}^{R_R}$ are situated precisely as in Case 3(a), and we merge representations and find private regions as before. See Figure 3.17.

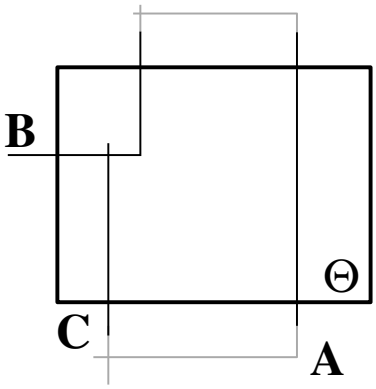


Figure 3.19: Completing a 3-sided $(int \cup (B, C))$ representation by adding intersections for (A, B) and (A, C).

This ends the description of the construction in all cases, and hence proves Lemma 3.14.

3.4 Extension from 4-connected triangulations to all planar graphs

In this section, we prove Theorem 3.1, i.e., we create B_2-VPG representations for all planar graphs. Observe that Lemma 3.14 essentially proves the theorem for 4-connected triangulations. As in [31] we extend the claim to hold for all triangulations by induction on the number of separating triangles.

Theorem 3.15. *Let G be a triangulation with outerface (A, B, C). G has a 1-string B_2-VPG representation with a chair-shaped private region for every interior face f of G.*

Proof. Our approach is exactly the same as in [31], except that we must be careful not to add too many bends when merging subgraphs at separating triangles, and hence must use 3-sided layouts. Formally, we proceed by induction on the number of separating triangles. In the base case, G has no separating triangle, i.e., it is 4-connected. As the outerface is a triangle, G clearly satisfies the chord condition. Thus, by Lemma 3.14, it has a 3-sided $(int \cup (B, C))$ representation R with private region for every face. R has an intersection for

every edge except for (A, B) and (A, C). These intersections can be created by tangling $\mathbf{B}, \mathbf{A}$ and $\mathbf{C}, \mathbf{A}$ suitably (see Figure 3.19). Recall that $\mathbf{A}$ initially did not have any bends, so it has 2 bends in the constructed representation of G. The existence of private regions is guaranteed by Lemma 3.14.

Now assume for the induction step that G has $k+1$ separating triangles. Let $\Delta = (a, b, c)$ be an inclusion-wise minimal separating triangle of G. Let G_2 be the graph induced by the vertices inside Δ, and let $G_1 = G - G_2$. Graph G_1 has k separating triangles. By induction, G_1 has a representation R_1 with a chair-shaped private region for every interior face f. Let Φ be the private region for face Δ. Permute a, b, c, if needed, so that the naming corresponds to the one needed for the private region and, in particular, the vertical segment of $\mathbf{c}$ intersects the private region of Δ as depicted in Figures 3.20 and 3.21. Chalopin et al. showed the following:

Observation 3.16 (Chalopin et al. [31])**.** *Subgraph G_2 is either an isolated vertex, or a W-triangulation with corners (A, B, C) such that the vertices on P_{AB} are adjacent to b, the vertices on P_{BC} are adjacent to c, and the vertices on P_{CA} are adjacent to a. Furthermore, G_2 satisfies the chord condition with respect to these corners.*

We can hence merge G_2 as follows:

Case 1: G_2 is a single vertex v. Represent v by inserting into Φ an orthogonal curve $\mathbf{v}$ with 2 bends that intersects $\mathbf{a}, \mathbf{b}$ and $\mathbf{c}$. The construction, together with private regions for the newly created faces (a, b, v), (a, c, v) and (b, c, v), is shown in Figure 3.20.

Case 2: G_2 is a W-triangulation. Recall that G_2 satisfies the chord condition with respect to corners (A, B, C). Apply Lemma 3.14 to construct a 3-sided $(int \cup (C, b_{s-1}))$ representation R_2 of G_2 with respect to the corners of G_2. Let us assume that (after possible rotation) Φ has the orientation shown in Figure 3.21; if it had the symmetric orientation then we would do a similar construction using a reverse 3-sided representation of G_2. Place R_2 inside Φ as shown in Figure 3.21. Stretch the curves representing vertices on P_{CA}, P_{AB} and $P_{Bb_{s-1}}$ downwards, upwards and leftwards respectively so that they intersect $\mathbf{a}, \mathbf{b}$ and $\mathbf{c}$. Top-tangle leftwards the curves $\mathbf{A} = \mathbf{a_1}, \mathbf{a_2}, \ldots, \mathbf{a_r} = \mathbf{B}$. Left-tangle downwards the curves $\mathbf{B} = \mathbf{b_1}, \mathbf{b_2}, \ldots, \mathbf{b_{s-1}}$ and bend and stretch $\mathbf{C}$ downwards so that it intersects

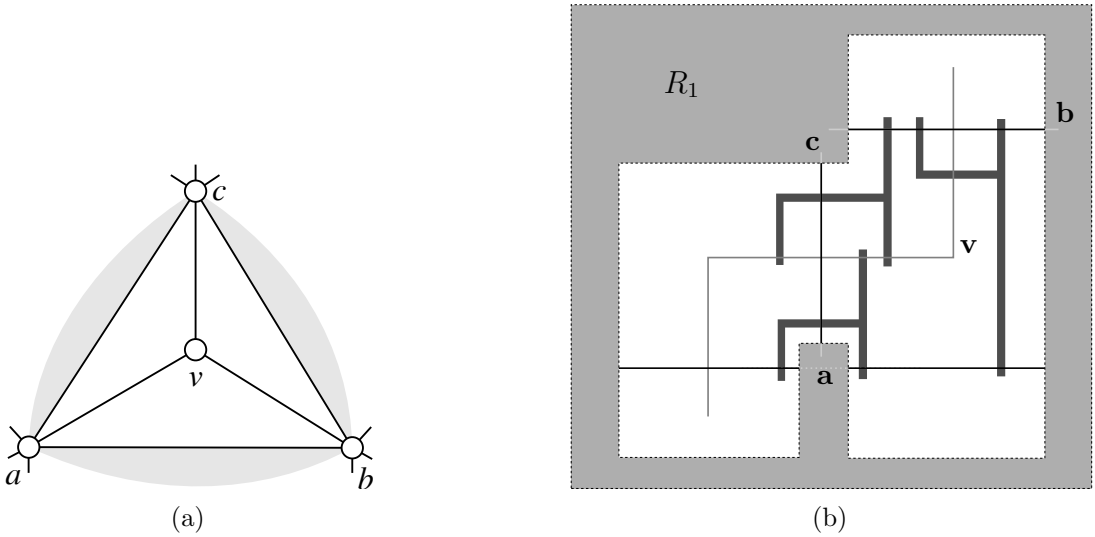


Figure 3.20: The construction for a separating triangle enclosing one vertex.

a. Bottom-tangle leftwards the curves $\mathbf{C} = \mathbf{c_1}, \ldots, \mathbf{c_t} = \mathbf{A}$. It is easy to verify that the construction creates intersections for all the edges between vertices of Δ and the outerface of G_2. The tangling operation then creates intersections for all the outerface edges of G_2 except edge (C, b_{s-1}), which is already represented in R_2.

Every curve that receives a new bend represents a vertex on the outerface of G_2, which means that it initially had at most 1 bend. Curve $\mathbf{A}$ is the only curve that receives 2 new bends, but this is allowed as $\mathbf{A}$ does not have any bends in R_2. Hence, the number of bends for every curve does not exceed 2.

Private regions for faces formed by vertices a, b, c and vertices on the outerface of G_2 can be found as shown in Figure 3.21. □

With Theorem 3.15 in hand, we can show our main result: every planar graph has a 1-string B_2-VPG representation.

Proof of Theorem 3.1. If G is a planar triangulated graph, then the claim holds by Theorem 3.15. To handle an arbitrary planar graph, repeatedly stellate the graph (recall that

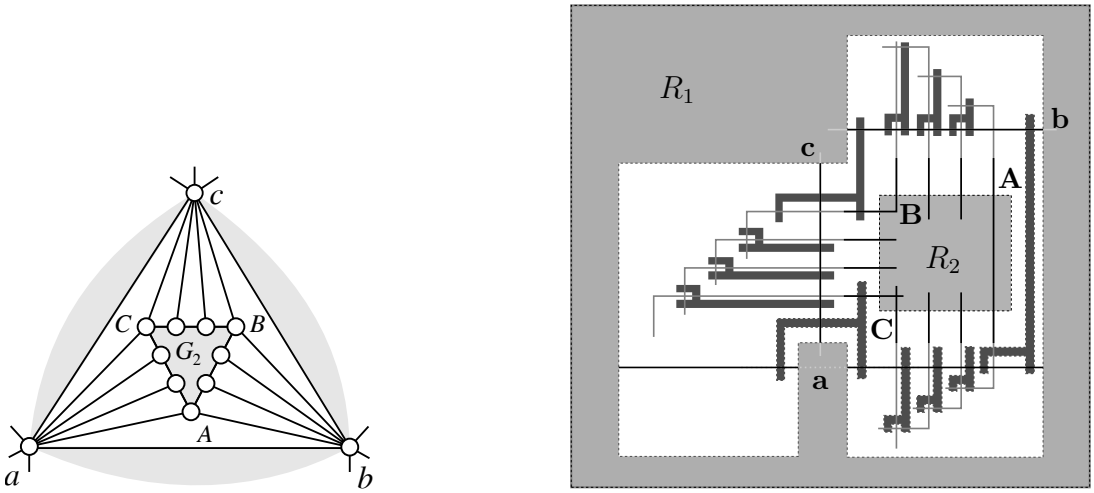


Figure 3.21: The construction for a separating triangle enclosing a W-triangulation.

this means inserting into each non-triangular face a new vertex connected to all vertices of the face). It is easily shown that one stellation makes the graph connected, a second one makes it 2-connected, and a third one makes it 3-connected and triangulated. Thus after 3 stellations we have a 3-connected triangulated graph G' such that G is an induced subgraph of G'. Apply Theorem 3.15 to construct a 1-string B_2-VPG representation R' of G' (with the three outerface vertices chosen as corners). By removing curves representing vertices that are not in G, we obtain a 1-string B_2-VPG representation of G. □

3.5 Example

Here we provide an example of constructing an $(int \cup (18, 16))$ 1-string B_2-VPG representation R of the W-triangulation shown in Figure 3.22. We use numbers and colors to distinguish vertices. We use letters to indicate special vertices such as corners; note that the designation as such a corner may change as the subgraph gets divided further. The special edge is marked with hatches.

One can verify that the graph with the chosen corners (1,4,18) satisfies the chord

condition. Vertex C has degree 4, but it is not incident to a chord, so one applies the construction from Section 3.3. Finding vertex $u_j = 6$, we can see that $j > 1$, so Case 3(a) applies. Figure 3.22 shows the graphs G_R, G_Q and G_0, and how to construct R from their representations R_R, R_Q and R_0.

The construction of R_Q is shown in Figure 3.23. The representation should have a 2-sided layout and no special edge. Graph G_Q decomposes into three subgraphs G_1, G_2, G_3. Their 2-sided representations are found separately (for G_1 this involves recursing twice more) and combined as described in the proof of Claim 3.13.

The construction of R_R is shown in Figure 3.24 (decomposition of G_R) and 3.25 (combining the representations). Representation R_R is supposed to be 3-sided. We first apply Case 1 (Section 3.3) twice, since corner C has degree 2. Then corner C becomes incident to a chord, so we are in Case 2, and use sub-case Case 2(a) (Section 3.3) since the special edge is $(C, b_{s-1} = B)$. This case calls for a 3-sided representation of a G_2 (which is a triangle in this case, so the base case applies). It also calls for a 2-sided representation of G_1 with special edge $(C, A = c_2)$. This is Case 2 (Section 3.2) and we need to apply the reversal trick—we flip the graph and relabel the corners. After obtaining the representation, it must be flipped horizontally in order to undo the reversal. The construction decomposes the graph further, using Case 2 repeatedly, which breaks the graphs into elementary triangles. Their 2-sided representations are obtained using the base case and composed as stipulated by the construction.

Figure 3.26 shows the complete 3-sided $(int \cup (18, 16))$ representation of the graph.

3.6 Conclusions

We showed that every planar graph has a 1-string B_2-VPG representation, i.e., a representation as an intersection graph of strings where strings cross at most once and each string is orthogonal with at most two bends.

Following the steps of our proof, it is not hard to see that our representation can be found in linear time, since the only non-local operation is to test whether a vertex has a

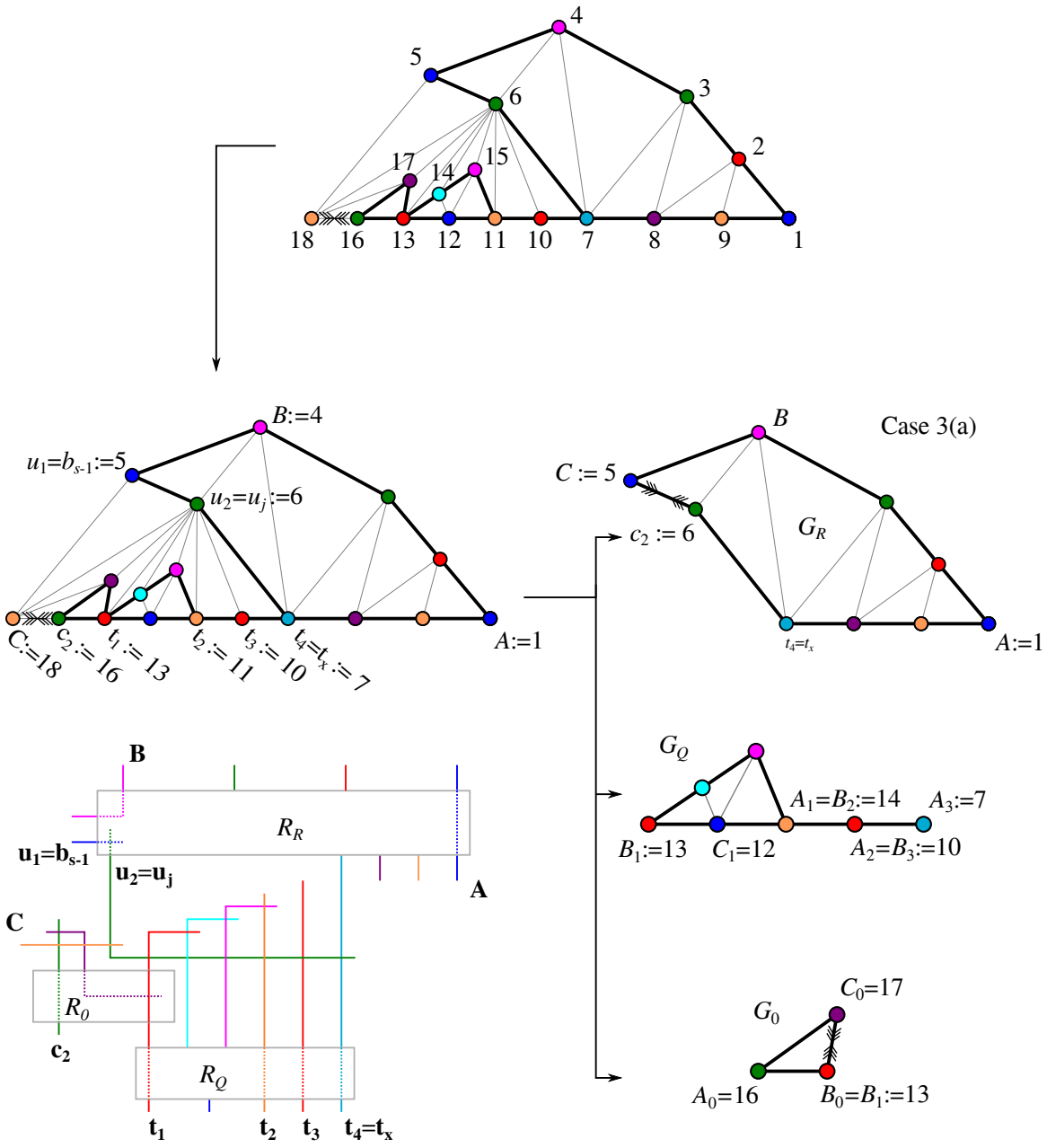


Figure 3.22: Illustration of the example. The goal is to find an $(int \cup (18, 16))$ 1-string B_2-VPG representation of the W-triangulation shown on top, using corners (1,4,18).

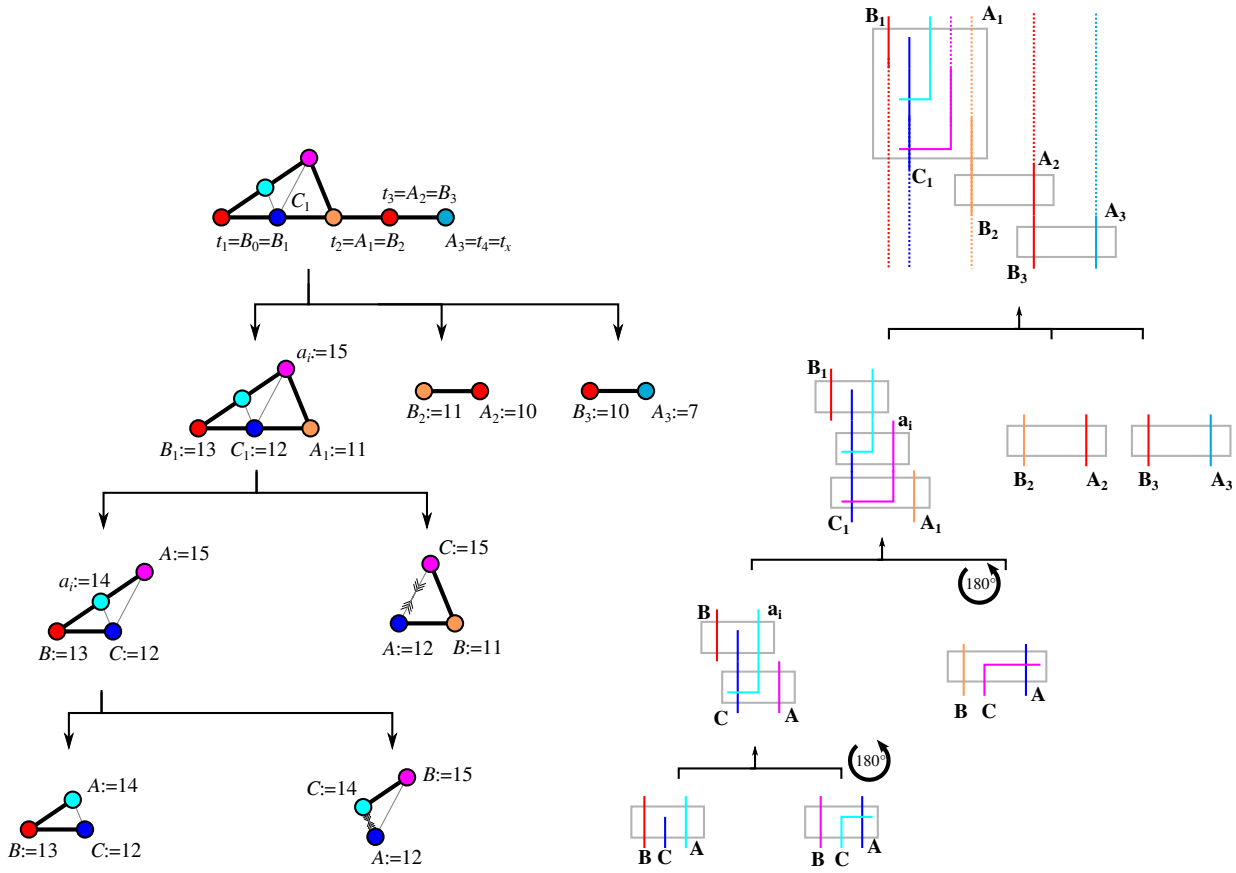


Figure 3.23: Illustration of the example: Finding R_Q (top right).

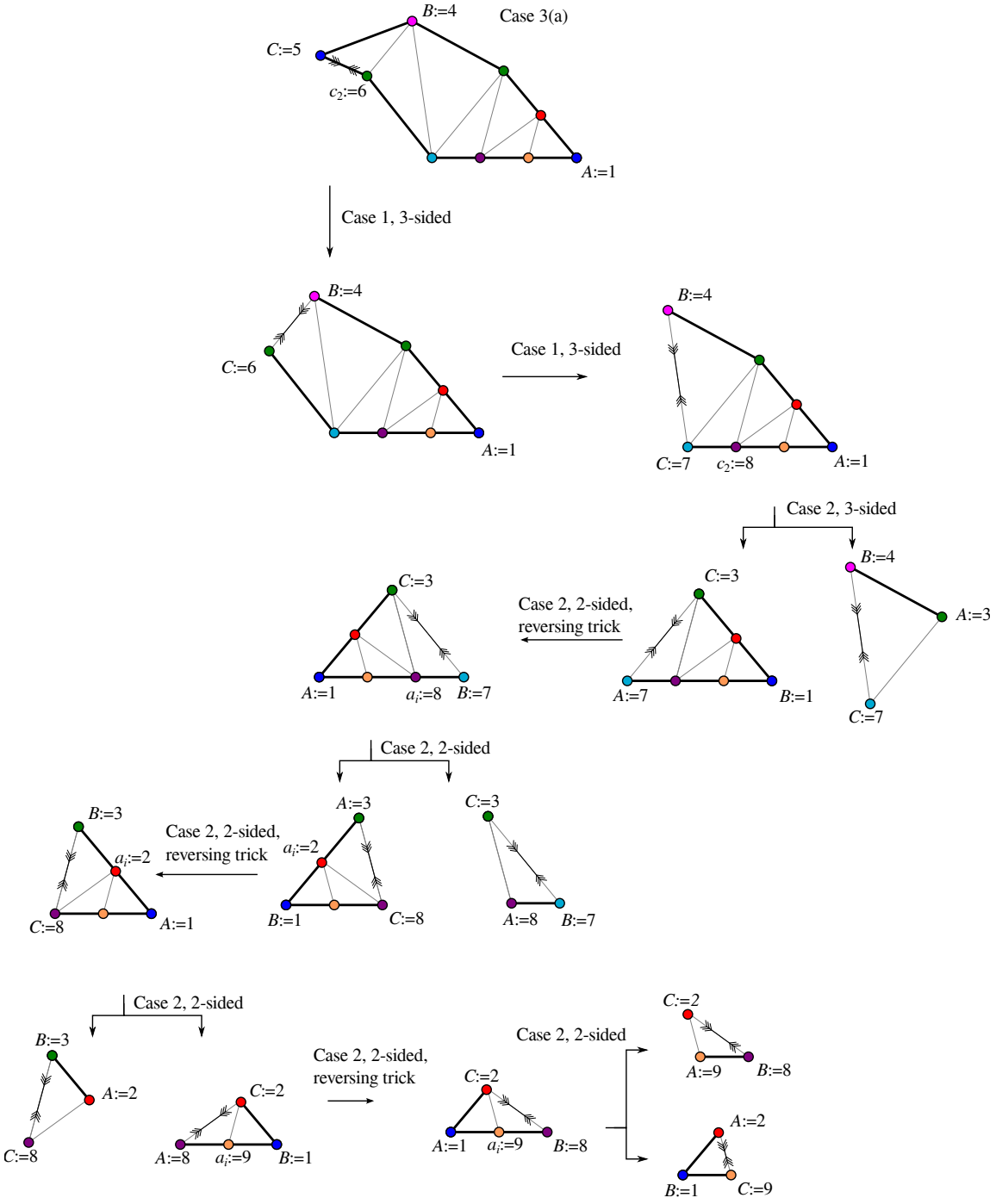


Figure 3.24: Illustration of the example: Decomposing graph G_R.

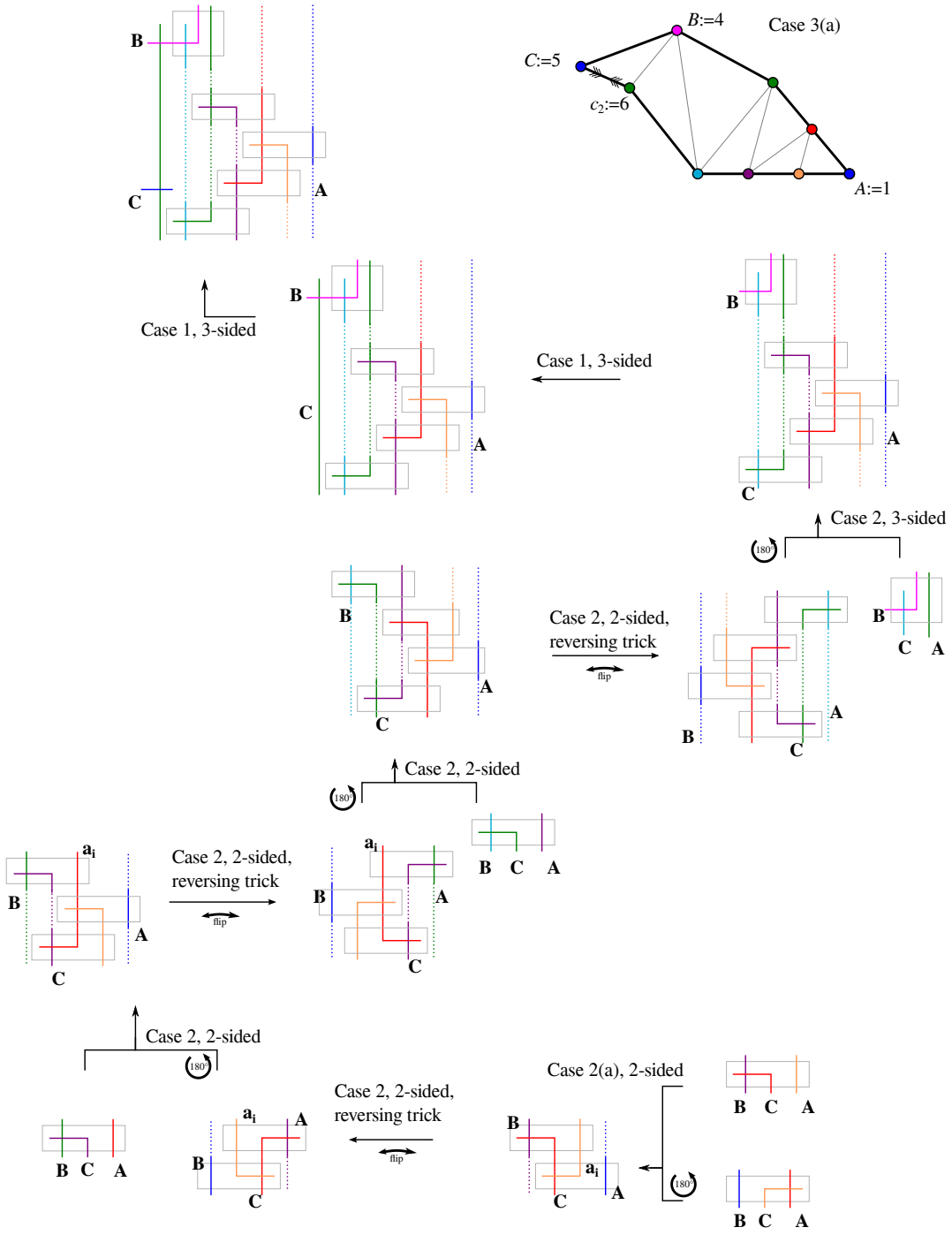


Figure 3.25: Illustration of the example: Composing representation R_R.

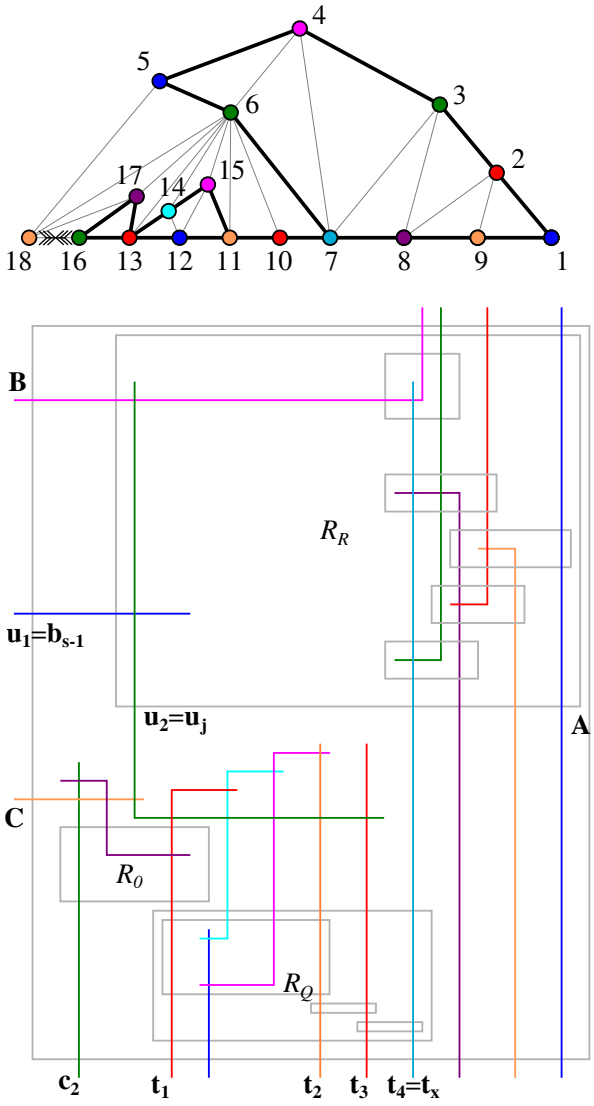


Figure 3.26: Illustration of the example: Complete 3-sided $(int \cup (18, 16))$ representation.

neighbour on the outerface. This can be tested by marking such neighbours whenever they become part of the outerface. Any vertex that becomes part of the outerface remains on the outerface throughout later steps, so such a marking happens only once per vertex. So this takes linear time overall.

The representation constructed in this paper uses curves of 8 possible shapes for planar graphs. For 4-connected planar graphs, the shapes that have at most one vertical segment suffice. A natural question is whether one can restrict the number of shapes required to represent all planar graphs, e.g., can we use only shapes ⌊, ⌞, ⌋ or ⌟ (those are the shape that suffice for 4-connected planar graphs) for all planar graphs?[3]

[3]Note that a very recent result of Gonçalves et al. [55] resolves this question positively. See also Section 8.2.

Chapter 4

Approximation Algorithms for B_1-VPG and B_2-VPG Graphs

In the previous chapter, we showed that one can construct 1-string B_2-VPG representations for planar graphs, and that a subset of such curves is sufficient to represent planar graphs that are 4-connected. We would now like to show that such representations are useful from an algorithmic point of view. This chapter is concerned with partitioning string graphs (and other classes of intersection graphs) into subgraphs that have nice properties, such as being outer-string graphs or permutation graphs (defined formally below). We can then use such a partition to obtain approximation algorithms for some graph problems, such as weighted independent set, clique, clique cover and colouring. More specifically, "partitioning" in this chapter usually means a *vertex partition*, i.e., we split the vertices of the graph as $V = V_1 \cup \cdots \cup V_k$ such that the subgraph induced by each V_i has nice properties. In one case we also do an *edge-partition* where we partition $E = E_1 \cup E_2$ and then work on the two subgraphs $G_i = (V, E_i)$, for $i = 1, 2$.

Our research was inspired by a paper by Lahiri et al. [71] (a similar technique was used earlier by Agarwal, van Kreveld and Suri in 1988 [2]). They gave an algorithm to approximate the maximum (unweighted) independent set in a B_1-VPG graph within a factor of $4\log^2 n$ (log in this thesis denotes $\log_2$). We greatly expand on their approach as follows. First, rather than solving maximum independent set directly, we instead split such

a B_1-VPG graph into subgraphs. This allows us to approximate not just independent set, but more generally any hereditary graph problem that is solvable in such graphs.

Secondly, rather than using co-comparability graphs for splitting as Lahiri et al. did, we use outer-string graphs. This allows us to stop the splitting earlier, reducing the approximation factor from $4\log^2 n$ to $2\log n$, and to give an algorithm for *weighted* independent set (wIS).

Finally, we allow much more general shapes, not just curves, for our intersection graphs. For splitting into outer-string graphs, we can allow any shape that can be described as the union of one vertical and any number of horizontal segments (we call such intersection graphs "single-vertical"). Our results yield a $2\log n$-approximation algorithm for wIS in such graphs, which include B_1-VPG graphs, and a $4\log n$-approximation for wIS in B_2-VPG graphs.

In the second part of the chapter, we consider splitting the graph such that the resulting subgraphs are co-comparability graphs. This type of problem was first considered by Keil and Stewart [62], who showed that so-called subtree filament graphs can be vertex-partitioned into $O(\log n)$ co-comparability graphs. The work of Lahiri et al. [71] can be seen as proving that every B_1-VPG graph can be vertex-partitioned into $O(\log^2 n)$ co-comparability graphs. We focus here on the bigger class of B_2-VPG graphs, and show that they can be vertex-partitioned into $O(\log^3 n)$ co-comparability graphs. Moreover, these co-comparability graphs have poset dimension 3, and if the B_2-VPG representation was 1-string, then they are permutation graphs. This leads to better approximation algorithms for clique, colouring and clique cover for B_2-VPG graphs.

The results of this chapter will appear in [15].

4.1 Decomposing into outer-string graphs

We argue in this section how to split a graph into outer-string graphs if it has an intersection representation of a special form. A *single-vertical object* is a connected set $S \subset \mathbb{R}^2$ of the form $S = s_0 \cup s_1 \cup \cdots \cup s_k$, where s_0 is a vertical segment and $s_1, \ldots, s_k$ are horizontal

segments, for some finite k. We consider a horizontal segment to be a single-vertical object as well, by attaching a zero-length vertical segment at one of its endpoints. Given a number of single-vertical objects $S_1, \ldots, S_n$, we define the intersection graph of them in the usual way, by defining one vertex per object and adding an edge whenever objects have at least one point in common. For the results in this section, it does not matter whether such a common point is a true crossing of segments; the approach works even if objects touch or overlap. We call such a representation a *single-vertical representation* and the graph a *single-vertical intersection graph* (see Figure 4.2 on page 72 for an example). The *x-coordinate* of one single-vertical object is defined to be the x-coordinate of the (unique) vertical segment. We will prove the following:

Theorem 4.1. *Let G be a single-vertical intersection graph. Then the vertices of G can be partitioned into at most* $\max\{1, 2\log n\}$ *sets*[1] *such that the subgraph induced by each is an outer-string graph.*

Our proof of Theorem 4.1 uses a splitting technique implicit in the recursive approximation algorithm of Lahiri et al. [71]. Let R be a single-vertical representation on G and let S be an ordered list of the x-coordinates of all the objects in R. We define the *median* m of R as the smallest number such that at most $\frac{|S|}{2}$ x-coordinates in S are smaller than m and at most $\frac{|S|}{2}$ x-coordinates in S are bigger than m. (If $|S|$ is odd then m is always the x-coordinate of at least one object.) Now split R into three sets: The *middle* set M of objects that intersect the vertical line $\mathbf{m}$ with x-coordinate m; the *left* set L of objects whose x-coordinates are smaller than m and that do not belong to M, and the *right* set R of objects whose x-coordinates are bigger than m and that do not belong to M. Split M further into $M_L = \{\ c \mid \text{the } x\text{-coordinate of } c \text{ is less than } m\}$ and $M_R = M \setminus M_L$. See also Figure 4.1.

Lemma 4.2. *The subgraph induced by the objects in M_L is outer-string.*

Proof. All the objects in M_L intersect curve $\mathbf{m}$. Since all the x-coordinates of those objects are smaller than m, all the intersections between two objects in M_L occur left of $\mathbf{m}$. For each $c \in M_L$, create a closed curve that traces around the part of c that is left of $\mathbf{m}$(by

[1] This bound is not tight; a more careful analysis shows that we get at most $\max\{1, 2\lceil\log n\rceil - 2\}$ sets.

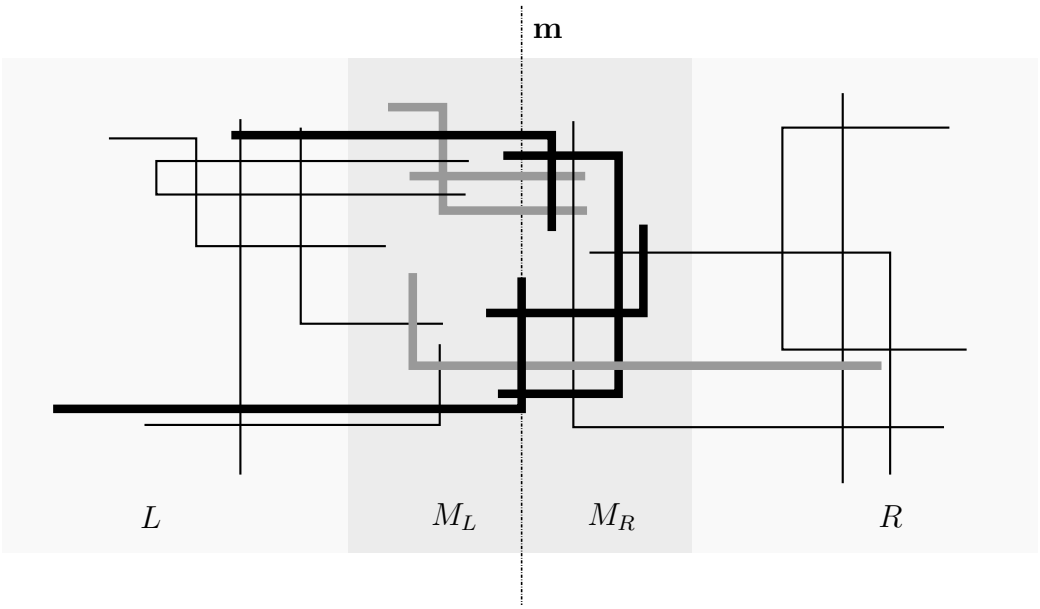


Figure 4.1: The split of a representation into L, $M = M_L \cup M_R$ and R.

tracing, we formally mean taking the boundary of a sufficiently small ϵ-neighbourhood of c that is left of **m**together with the segment of **m**that belongs to this ϵ-neighbourhood). Breaking the closed trace-curve at one of the attachments to **m** produces an open curve. Doing so for every object, one obtains an outer-string representation where all curves attach to **m** from one side and that induces the same graph as M_L. □

A similar proof shows that the graph induced by objects in M_R is an outer-string graph. Now we can prove our main result:

Proof of Theorem 4.1. Let G be a graph with a single-vertical representation. We proceed by induction on the number of vertices n in G. If $n \leq 2$, then the graph is outer-string and we are done, so assume $n \geq 3$, which implies that $\log n \geq \frac{3}{2}$. By Lemma 4.2, both M_L and M_R individually induce an outer-string graph. Applying induction, we get at most

$$\max\{1, 2\log|L|\} \leq \max\{1, 2\log(n/2)\} = \max\{1, 2\log n - 2\} = 2\log n - 2$$

outer-string subgraphs for L, and similarly at most $2\log n - 2$ outer-string subgraphs for R. Since the objects in L and R are separated by the vertical line **m**, there are no edges

between the corresponding vertices. Thus any outer-string subgraph defined by L can be combined with any outer-string subgraph defined by R to give one outer-string graph. We hence obtain $2\log n - 2$ outer-string graphs from recursing into L and R. Adding to this the two outer-string graphs defined by M_L and M_R gives the result. □

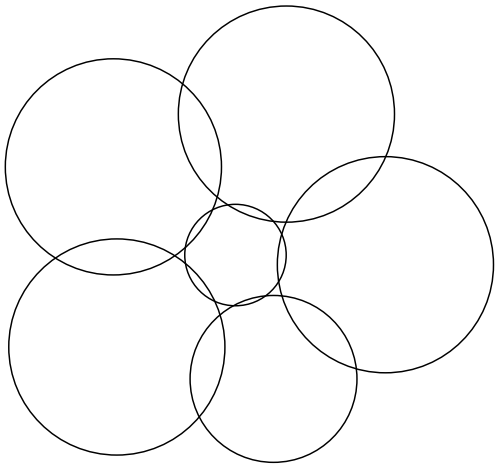
Our proof is constructive, and finds the partition within $O(\log n)$ recursions. In each recursion we must find the median m and then determine which objects intersect the line **m**. If we pre-sort three lists of the objects (once by x-coordinate of the vertical segment, once by leftmost x-coordinate, and once by rightmost x-coordinate), and pass these lists along as parameters, then each recursion can be done in $O(n)$ time, without linear-time median-finding. The pre-sorting takes $O(N + n\log n)$ time, where N is the total number of segments in the representation. Hence the run-time to find the partition into $O(\log n)$ outer-string graphs is $O(N + n\log n)$.

4.1.1 What graphs are single-vertical?

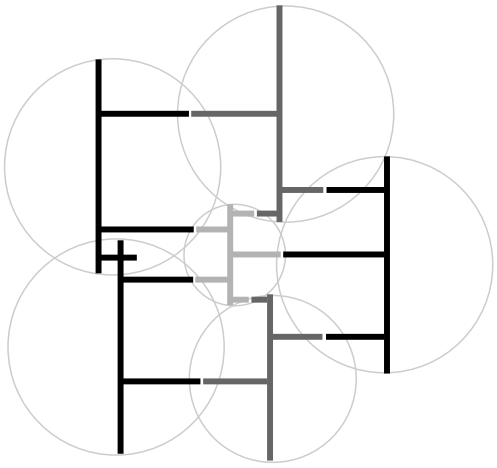
The above results were for single-vertical graphs. However, the main focus of this chapter is B_k-VPG graphs, for $k \leq 2$. Clearly B_1-VPG graphs are single-vertical by definition. It is not obvious whether all B_2-VPG graphs are single-vertical graphs. Note that a B_2-VPG representation may not be a single-vertical representation—it may have curves with two horizontal segments as well as curves with two vertical segments, so no rotation of the representation can give a single-vertical representation. However, we can still handle them by doubling the number of graphs into which we split.

Lemma 4.3. *Let G be a B_2-VPG graph. Then the vertices of G can be partitioned into 2 sets such that the subgraph induced by each is a single-vertical B_2-VPG graph.*

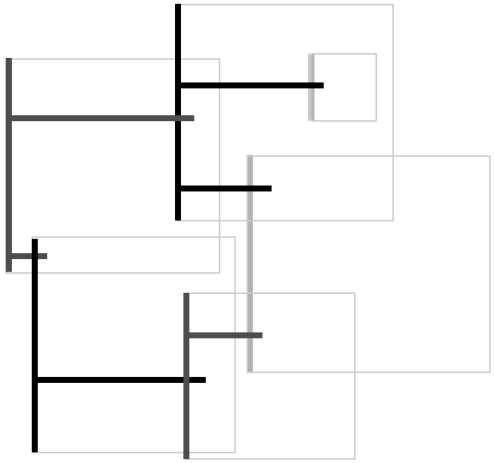
Proof. Fix a B_2-VPG representation of G. Let V_v be the vertices that have at most one vertical segment in their curve, and V_h be the remaining vertices. Since every curve has at most three segments, and all curves in V_h have at least two vertical segments, each of them has at most one horizontal segment. Clearly V_v induces a single-vertical graph. V_h *also* induces a single-vertical graph, because we can rotate all curves by 90° and then have at most one vertical segment per curve. □



(a) A disk graph. The graph corresponds to the graph in Figure 2.1.



(b) A single-vertical representation obtained from a disk graph.



(c) A single-vertical representation obtained from a boxicity-2 graph.

Figure 4.2: An example of a single-vertical representation.

Combining this with Theorem 4.1, we immediately obtain:

Corollary 4.4. *Let G be a B_2-VPG graph. Then the vertices of G can be partitioned into at most $\max\{1, 4\log n\}$ sets such that the subgraph induced by each is an outer-string graph.*

In particular, by Theorem 3.1, all planar graphs are B_2-VPG graphs and Corollary 4.4 applies to them. A number of graph classes can also be shown to be subclasses of single-vertical graphs. In particular, this includes boxicity-2 graphs (intersection graphs of axis-aligned rectangles in the plane) and disk graphs (intersection graphs of circles in the plane)—we can replace these shapes (rectangles or disks) by a vertical segment and sufficiently many horizontal segments to cover at east one common point for each intersecting point of shapes (see Figure 4.2). Finally, there exist a generalization of planar graph called 1-*planar graphs*. These also turn out to be single-vertical graphs under some conditions on crossing edges. We return to this in Section 7.3.

4.2 Decomposing into co-comparability graphs

We now show that by doing further splits, we can actually decompose B_2-VPG graphs into so-called co-comparability graphs of poset dimension 3 (defined formally below). While we require more subgraphs for such a split, the advantage is that numerous problems can be solved in polynomial time for co-comparability graphs, while for outer-string graphs we know of no problem other than weighted independent set that is poly-time solvable.

We first give an outline of the approach. Given a B_2-VPG graph, we first use Lemma 4.3 to split it into two single-vertical B_2-VPG graphs. Given a single-vertical B_2-VPG graph, we next use a technique much like the one of Theorem 4.1 to split it into $\log n$ single-vertical B_2-VPG graphs that are "centered" in some sense. Any such graph can easily be edge-partitioned into two B_1-VPG graphs that are "grounded" in some sense. We then apply the technique of Theorem 4.1 again (but in the other direction) to split a grounded B_1-VPG graph into $\log n$ B_1-VPG graphs that are "cornered" in some sense. The latter graphs can be shown to be permutation graphs. This gives the result after arguing that the edge-partition can be un-done at the cost of combining permutation graphs into co-comparability graphs.

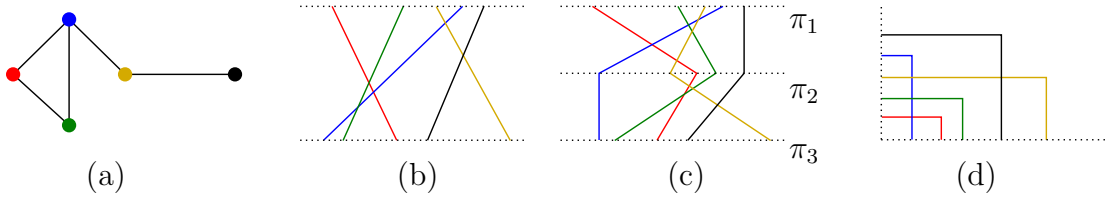


Figure 4.3: (a) A graph that is simultaneously (b) a permutation graph; (c) a co-comparability graph of poset dimension 3; and (d) a cornered B_1-VPG graph.

For this section, we return to the usual assumption that any intersection of strings is a true intersection, i.e., neither a touching nor an overlap. Recall that by Lemma 2.4, we may hence assume that x-coordinates and y-coordinates of segments are distinct. We may also assume that every vertex is represented by an object with *exactly* 3 segments—we can achieve this by attaching very short segments that intersect nothing.

4.2.1 Co-comparability graphs

We start by defining the graph classes that we use. A graph G with vertices $\{1, \ldots, n\}$ is called a *permutation graph* if there exist two permutations π_1, π_2 of $\{1, \ldots, n\}$ such that (i, j) is an edge of G if and only if π_1 lists i, j in the opposite order as π_2 does. Put differently, if we place $\pi_1(1), \ldots, \pi_1(n)$ at points along a horizontal line, and $\pi_2(1), \ldots, \pi_2(n)$ at points along a parallel horizontal line, and use the line segment $(\pi_1(i), \pi_2(i))$ to represent vertex i, then the graph is the intersection graph of these segments. See Figure 4.3(b).

A *co-comparability graph* G is a graph whose complement can be directed in an acyclic transitive fashion. Rather than defining these terms, we describe here only the restricted type of co-comparability graphs that we are interested in. A graph G with vertices $\{1, \ldots, n\}$ is called a *co-comparability graph of poset dimension* k if there exist k permutations $\pi_1, \ldots, \pi_k$ such that (i, j) is an edge if and only if there are two permutations that list i and j in opposite order. See Figure 4.3(c) and refer to Golumbic et al. [54] for more on these characterizations. Note that a permutation graph is a co-comparability graph of poset dimension 2.

4.2.2 Cornered B_1-VPG graphs

A B_1-VPG representation is called *cornered* if there exists a horizontal and a vertical ray emanating from the same point such that any curve of the representation intersects both rays. See Figure 4.3(d) for an example.

Lemma 4.5. *If G has a cornered B_1-VPG representation, say with respect to rays r_1 and r_2, then G is a permutation graph. Further, the two permutations defining G are exactly the two orders in which vertex-curves intersect r_1 and r_2.*

Proof. Since the curves have only one bend, the intersections with r_1 and r_2 determine the curve of each vertex. In particular, two curves intersect if and only if the two orders along r_1 and r_2 are *not* the same, which is to say, if their orders are different in the two permutations of the vertices defined by the orders along the rays. Hence using these orders shows that G is a permutation graph. □

4.2.3 From grounded to cornered

We call a B_1-VPG representation *grounded*[2] if there exists a horizontal line segment ℓ_H that intersects all curves, and has all horizontal segments of all curves strictly above it. See also Figure 4.4. We now show how to split a grounded B_1-VPG representation into cornered ones. It will be important later that not only can we do such a split, but we know how the curves intersect ℓ_H afterwards. More precisely, the curves in the resulting representations may not be identical to the ones we started with, but they are modified only in such a way that the intersections points of curves along ℓ_H is unchanged.

Lemma 4.6. *Let R be a B_1-VPG representation that is grounded with respect to segment ℓ_H. Then R can be partitioned into at most $\max\{1, 2\log n\}$ sets $R_1, \ldots, R_K$ such that each set R_i is cornered after upward translation and segment-extension of some of its curves.*

Proof. A single curve with one bend is always cornered, so the claim is easily shown for $n \leq 4$ where $\max\{1, 2\log n\} \geq n$. For $n \geq 5$, it will be helpful to split R first into two sets,

[2]The term "grounded" has been used for other types of intersection graphs previously, see e.g. [26, 27].

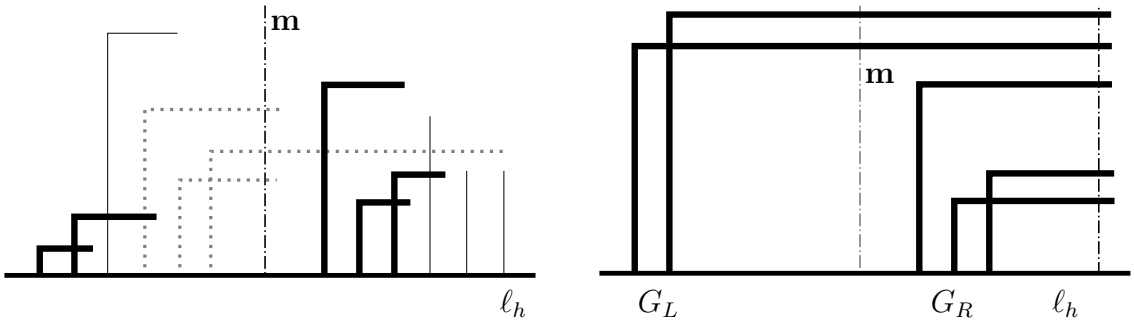


Figure 4.4: An illustration for the proof of Lemma 4.6. (a) Splitting a cornered B_1-VPG graph. The dotted curves form a cornered B_1-VPG graph and the algorithm recurses in the solid curved. (b) Combining a graph G_L with a graph G_R found in the recursive step (the bold curves in (a)) so that the result is a cornered B_1-VPG graph.

those curves that form ⌜ and those that form ⌝ (no other shapes can exist in a grounded B_1-VPG representation). The result follows if we show the following:

Claim. A grounded B_1-VPG representation that consists only of shapes ⌜ can be split into $\log n$ many cornered B_1-VPG representations.

So assume that R consists of only ⌜'s. We apply essentially the same idea as in Theorem 4.1. Let again **m** be the vertical line along the median of x-coordinates of vertical segments of curves. Let M be all those curves that intersect **m**. Let **m'** be the vertical line just to the right of **m**. After extending horizontal segments of M a bit (if needed) all curves in M intersect **m'**, and none of them has the vertical segment on **m'**. Since curves are ⌜'s, any curve in M intersects ℓ_H to the left of **m'**, and intersects **m'** above ℓ_H. Hence taking the two rays along ℓ_H and **m'** emanating from their common point shows that M is cornered.

We then recurse both in the subgraph L of vertices whose curves lie entirely left of **m** and the subgraph R of vertices whose curves lie entirely right of **m**. Each of them is split recursively into at most $\max\{1, \log(n/2)\} = \log n - 1$ subgraphs that are cornered. We currently have twice as many subgraphs as required, so we must show how to combine two such subgraphs G_L and G_R (of vertices from L and R) such that they are cornered while

modifying curves only in the permitted way.

Translate curves of G_L upward such that the lowest horizontal segment of G_L is above the highest horizontal segment of G_R. Extend the vertical segments of G_L so that they again intersect ℓ_H. Extend horizontal segments of both G_L and G_R rightward until they all intersect one vertical line segment. See also Figure 4.4b. The resulting representation satisfies all conditions.

Since we obtain at most $\log n - 1$ such cornered representations from the curves in $R \cup L$, we can add M to it and the result follows. □

Corollary 4.7. *Let G be a graph with a grounded B_1-VPG representation. Then the vertices of G can be partitioned into at most $\max\{1, 2\log n\}$ sets such that the subgraph induced by each is a permutation graph.*

4.2.4 From centered to grounded

We now switch to VPG representations with 2 bends, but currently only allow those with a single vertical segment per curve. So let R be a single-vertical B_2-VPG representation. We call R *centered* if there exists a horizontal line segment ℓ_H that intersects the vertical segment of all curves in its interior. Given such a representation, we can cut each curve apart at the intersection point with ℓ_H. Then the parts above ℓ_H form a grounded B_1-VPG representation, and the parts below form (after a 180° rotation) also a grounded B_1-VPG representation. Note that this split corresponds to splitting the edges into $E = E_1 \cup E_2$, depending on whether the intersection for each edge occurs above or below ℓ_H. If curves may intersect repeatedly, then an edge may be in both sets. See Figure 4.5 for an example. With this, we can now split into co-comparability graphs.

Lemma 4.8. *Let G be a graph with a single-vertical centered B_2-VPG representation. Then the vertices of G can be partitioned into at most $\max\{1, 4\log^2 n\}$ sets such that the subgraph induced by each is a co-comparability graph of poset dimension 3.*

Proof. The claim clearly holds for $n \leq 4$, so assume $n \geq 5$. Let ℓ_H be the horizontal segment along which the representation is centered. Split the edges into $E = E_1 \cup E_2$ as

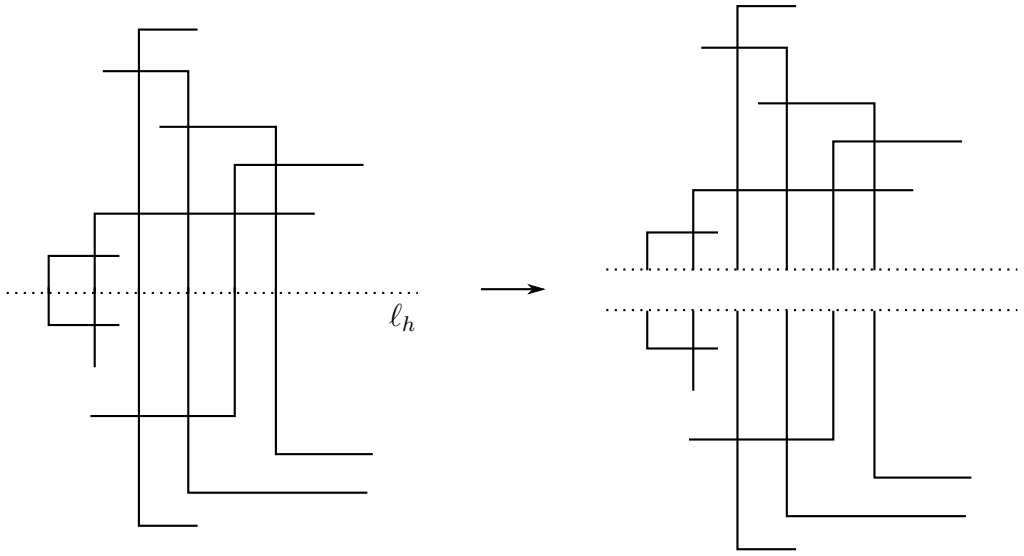


Figure 4.5: Splitting a single-vertical B_2-VPG representation into two grounded B_1-VPG representations.

above, and let R_1 and R_2 be the resulting grounded B_1-VPG representations, which have the same order of vertical intersections along ℓ_H. Split R_1 into $K \leq 2\log n$ sets of curves $R_1^1, \ldots, R_1^K$, each of which forms a cornered B_1-VPG representation that uses the same order of intersections along ℓ_H. Similarly split R_2 into $K' \leq 2\log n$ sets $R_2^1, \ldots, R_2^{K'}$ of cornered B_1-VPG representations.

Now define $R_{i,j}$ to be all those curves r in R where the part of r above ℓ_H belongs to R_1^i and the part below belongs to R_2^j. This gives $K \cdot K' \leq 4\log^2 n$ sets of curves. Consider one such set $R_{i,j}$. The parts of curves in $R_{i,j}$ that were above ℓ_H are cornered at ℓ_H and some vertical upward ray, hence define a permutation π_1 along the vertical ray and π_2 along ℓ_H. Similarly the parts of curves below ℓ_H define two permutations, say π_2' along ℓ_H and π_3 along some vertical downward ray. But the split into cornered B_1-VPG representation ensured that the intersections along ℓ_H was not changed, so $\pi_2 = \pi_2'$. The three permutations π_1, π_2, π_3 together hence define a co-comparability graph of poset dimension 3 as desired. □

We can do slightly better if the representation is additionally 1-string.

Corollary 4.9. *Let G be a graph with a single-vertical centered 1-string B_2-VPG representation. Then the vertices of G can be partitioned into at most $\max\{1, 4\log^2 n\}$ sets such that the subgraph induced by each is a permutation graph.*

Proof. The split is exactly the same as in Lemma 4.8. Consider one of the subgraphs G_i and the permutations π_1, π_2, π_3 that came with it, where π_2 is the permutation of curves along the centering line ℓ_H. We claim that G_i is a permutation graph, using π_1, π_3 as the two permutations. Clearly if (u, v) is not an edge of G_i, then all of π_1, π_2, π_3 list u and v in the same order. If (u, v) is an edge of G_i, then two of π_1, π_2, π_3 list u, v in opposite order. We claim that π_1 and π_3 list u, v in opposite order. For if not, say u comes before v in both π_1 and π_3, then (to represent edge (u, v)) we must have u after v in π_2. But then the curves of u and v intersect both above and below ℓ_H, contradicting that we have a 1-string representation. So the two permutations π_1, π_3 define graph G_i. □

4.2.5 Making single-vertical B_2-VPG representations centered

Lemma 4.10. *Let G be a graph with a single-vertical B_2-VPG representation. Then the vertices of G can be partitioned into at most $\max\{1, \log n\}$ sets such that the subgraph induced by each has a single-vertical centered B_2-VPG representation.*

Proof. The approach is quite similar to the one in Theorem 4.1, but uses a horizontal split and a different median. The claim is easy to show for $n \leq 3$, so assume $n \geq 4$. Recall that there are n vertical segments, hence $2n$ endpoints of such segments. We assumed that no two horizontal segments have the same y-coordinate, so the $2n$ endpoints of vertical segments give $2n$ distinct y-coordinates. Let m be a value such that exactly n of these endpoints are strictly below m and exactly n are strictly above m, and let $\mathbf{m}$ be the horizontal line with y-coordinate m.

Let M be the curves that are intersected by $\mathbf{m}$; clearly they form a single-vertical centered B_2-VPG representation. Let B be all those curves whose vertical segment (and hence the entire curve) is completely below $\mathbf{m}$. Each such curve contributes two endpoints of vertical segments, hence $|B| \leq n/2$ by choice of m. Recursively split B into at most

$\max\{1, \log(n/2)\} = \log n - 1$ sets, and likewise split the curves U above $\mathbf{m}$ into at most $\log n - 1$ sets.

Each chosen subset R_B of B is centered, as is each subset R_U of U chosen in the recursive step matching R_B. Since R_B uses curves below $\mathbf{m}$ while R_U uses curves above, there are no crossings between these curves. We can hence translate the curves of R_B horizontally and vertically so that they are centered with the same horizontal line as R_U. Therefore $R_B \cup R_U$ has a centered single-vertical B_2-VPG representation. Repeating this for all of $B \cup U$ gives $\log n - 1$ centered single-vertical B_2-VPG representations, to which we can add the one defined by M. □

4.2.6 Putting it all together

We summarize all these results in our main result about splits into co-comparability graphs:

Theorem 4.11. *Let G be a B_2-VPG graph. Then the vertices of G can be partitioned into at most $\max\{1, 8\log^3 n\}$ sets such that the subgraph induced by each is a co-comparability graph of poset dimension 3. If G is a 1-string B_2-VPG graph, then the subgraphs are permutation graphs.*

Proof. The claim is trivial for $n = 1$ and holds for $n = 2, 3$ since then $n \leq 8\log^3 n$, so assume $n \geq 4$. Fix a B_2-VPG representation R. First split R into two single-vertical B_2-VPG representations as in Lemma 4.3. Split each of them into $\log n$ single-vertical centered B_2-VPG representations using Lemma 4.10, for a total of at most $2\log n$ sets of curves. Split each of them into $4\log^2 n$ co-comparability graphs (or permutation graphs if the representation was 1-string) using Lemma 4.8 or Corollary 4.9. The result follows. □

We can do better for B_1-VPG graphs. The subgraphs obtained in the result below are the same ones that were used implicitly in the $4\log^2 n$-approximation algorithm given by Lahiri et al. [71].

Theorem 4.12. *Let G be a B_1-VPG graph. Then the vertices of G can be partitioned into at most $\max\{1, 4\log^2 n\}$ sets such that the subgraph induced by each is a permutation graph.*

Proof. The claim is trivial if $n = 1$, so assume $n > 1$. Fix a B_1-VPG representation R, and split it into $\log n$ single-vertical centered B_1-VPG representations using Lemma 4.10. Split each of them into two centered B_1-VPG representations, one of those curves with the horizontal segment above the centering line, and one with the rest. Each of the resulting $2 \log n$ centered B_1-VPG representations is now grounded (possibly after a 180° rotation). We can split each of them into $2 \log n$ permutation graphs using Corollary 4.7, for a total of $4 \log^2 n$ permutation graphs. □

4.3 Applications

We now show how Theorem 4.1 and 4.14 can be used for improved approximation algorithms for B_2-VPG graphs. The techniques used here are virtually the same as the one by Keil and Stewart [62] and require two things. First, the problem considered needs to be solvable on the special graph class (such as outer-string graphs or co-comparability graphs or permutation graphs) that we use. Second, the problem must be *hereditary* in the sense that a solution in a graph implies a solution in an induced subgraph, and solutions in induced subgraphs can be used to obtain a decent solution in the original graph.

We demonstrate this in detail using weighted independent set, which Keil et al. showed to be polynomial-time solvable in outer-string graphs [63, 61]. Recall that this is the problem: given a graph with vertex-weights, find a subset I of vertices that has no edges between them such that $w(I) := \sum_{v \in I} w(v)$ is maximized, where $w(v)$ denotes the weight of vertex v. The run-time to solve weighted independent set in outer-string graphs is $O(N^3)$, where N is the number of segments in the given outer-string representation.

Theorem 4.13. *There exists a $(2 \log n)$-approximation algorithm for weighted independent set on single-vertical graphs with run-time $O(N^3)$, where N is the total number of segments used among all single-vertical objects.*

Proof. If $n = 1$, then the unique vertex is the maximum weight independent set. Else, use Theorem 4.1 to partition the vertices of the given graph G into at most $2 \log n$ sets, each of which induces an outer-string graph, and find the largest weighted independent set in

each applying the algorithm of Keil et al. If G_i had an outer-string representation with N_i segments in total, then this takes time $O(\sum N_i^3)$ time. Note that if a single-vertical object consisted of one vertical and ℓ horizontal segments, then we can trace around it with a curve with $O(\ell)$ segments. Hence all curves together have $O(N)$ segments and the total run-time is $O(N^3)$ which dominates the $O(N + n \log n)$ time needed to partition the vertices.

Let I_i^* be the maximum-weight independent set in G_i, and return as set I the set in $I_1^*, \ldots, I_k^*$ that has the maximum weight. To argue the approximation-factor, let I^* be the maximum-weight independent set of G, and define I_i to be all those vertices of I^* whose representation belongs to R_i, for $i = 1, \ldots, k$. Clearly I_i is an independent set of G_i, and so $w(I_i) \leq w(I_i^*)$. But on the other hand $\max_i\{w(I_i)\} \geq w(I^*)/k$ since we split I^* into k sets. Therefore $w(I) = \max_i\{w(I_i^*)\} \geq w(I^*)/k$, and so the returned independent set is within a factor of $k \leq 2 \log n$ of the optimum. □

We note here that the best algorithm for independent set in general string graphs achieves an approximation factor of $O(n^\varepsilon)$, under the assumption that any two strings cross each other at most a constant number of times [46]. This algorithm only works for unweighted independent set; we are not aware of any approximation results for weighted independent set in arbitrary string graphs.

Because B_2-VPG graphs can be vertex-split into two single-vertical B_2-VPG representations, and the total number of segments used is $O(n)$, we also get:

Corollary 4.14. *There exists a $(4 \log n)$-approximation algorithm for weighted independent set on B_2-VPG graphs with run-time $O(n^3)$.*

Another hereditary problem is *colouring*: Find the minimum number k such that we can assign numbers in $\{1, \ldots, k\}$ to vertices such that no two adjacent vertices receive the same number. Fox and Pach [46] pointed out that if we have a c-approximation algorithm for Independent Set, then we can use it to obtain an $O(c \log n)$-approximation algorithm for colouring. Therefore our result also immediately implies an $O(\log^2 n)$-approximation algorithm for colouring in single-vertical graphs and B_2-VPG graphs.

Another hereditary problem is *weighted clique*: Find the maximum-weight subset of vertices such that any two of them are adjacent. (This is independent set in the complement graph.) Clique is NP-hard in outer-string graphs even in its unweighted version [24]. For this reason, we use the split into co-comparability graphs instead; weighted clique can be solved in quadratic time in co-comparability graphs (because weighted independent set is linear-time solvable in comparability graphs [53]). Weighted clique is also linear-time solvable on permutation graphs [53]. We therefore have:

Theorem 4.15. *There exists an* $(8\log^3 n)$*-approximation algorithm for weighted clique on* B_2*-VPG graphs with run-time* $O(n^2)$*. The run-time becomes* $O(n)$ *if the graph is a 1-string* B_2*-VPG graph, and the approximation factor becomes* $4\log^2 n$ *if the graph is a* B_1*-VPG graph.*

A similar result holds for *clique cover*, which is the problem of colouring the complement: Find the minimum k such that the vertex set can be partitioned into k sets, each of which induces a clique. The complexity of clique cover is unknown for outer-string graphs, but it is polynomial for co-comparability graphs [62].

Theorem 4.16. *There exists a* $(8\log^3 n)$*-approximation algorithm for clique cover on* B_2*-VPG graphs with run-time* $O(n^2)$*.*

In a similar manner, we can get poly-time $(8\log^3 n)$-approximation algorithms for any hereditary problem that is solvable on co-comparability graphs. This includes maximum k-colourable subgraph and maximum h-coverable subgraph. See [62] for the definition of these problems, and the argument that they are hereditary and polynomial in co-comparability graphs.

4.4 Conclusions

In this chapter, we presented a technique for decomposing single-vertical graphs into outer-string subgraphs, B_2-VPG graphs into co-comparability graphs, and 1-string B_2-VPG

graphs into permutation graphs. We then used these results to obtain approximation algorithms for hereditary problems, such as weighted independent set.

As for open problems, we are very interested in approximation algorithms for B_k-VPG graphs, where k is a constant. Also, if curves are not required to be orthogonal, but have few bends, are there approximation algorithms better than those for arbitrary string graphs?

Chapter 5

B_1-VPG Representations

In Chapter 3, we showed that every planar graph has a 1-string B_2-VPG representation. Ideally, we would like a B_1-VPG representation instead. Whether this exists for all planar graphs remains one of the big open questions in the field[1]. In this chapter, we investigate subclasses of planar graphs that do not require two bends in their 1-string representations, i.e., that are B_1-VPG graphs. Some of the results of this chapter were published in [11].

Note that strings in B_1-VPG representations have 6 possible shapes ⌞, ⌜, ⌟, ⌝, |, -. We use the notation of an {⌞}-*representation* to denote a representation where every shape is an ⌞, and similarly for other subsets of {⌞,⌜,⌟,⌝,|,-}. Note that shapes | and - in string representations can always be tranformed into any other shape with 1 bend (this is not generally true in the contact representations that we describe below).

In this chapter, we will several times conside so-called contact representations as they can be used to produce string representations, and the classes of graphs with contact representations have been extensively studied. We say that two curves make *contact* if the endpoint of one of them coincides with an interior point of the other. Sometimes, we allow two curves to end at the same point as long as no third curve uses this point; this becomes a contact after extending one curve a bit. A *contact representation* of a graph $G = (V, E)$

[1]Note that a very recent result of Gonçalves et al. [55] resolves this question positively. See also Section 8.2.

is an arrangement of non-crossing curves $\mathbf{v}$ for each $v \in V$, such that $\mathbf{u}$ and $\mathbf{v}$ make contact if and only if $(u, v) \in E$. By stretching every curve by a small amount, one can turn every contact into an intersection without creating any new intersections. Thus, every graph class $\mathcal{S}$ that has a contact representation that uses curves admissible in B_1-VPG representations is a subclass of B_1-VPG. Let us point out that any such class $\mathcal{S}$ is strictly smaller than B_1-VPG as all graphs in $\mathcal{S}$ must be planar, but any arbitrarily large clique (including K_5) belongs to B_1-VPG. We use $\{\llcorner\}$-*contact representation* to denote a contact representation where every shape is an $\llcorner$, and similarly for other subsets of $\{\llcorner, \ulcorner, \lrcorner, \urcorner, |, -\}$. For brevity, we refer to $\{\llcorner, \ulcorner, \lrcorner, \urcorner, |, -\}$-contact representations as B_1-VPG-contact representations. A graph is a B_1-*VPG-contact* graph if it has a B_1-VPG-contact representation. Note that in contact representations, shapes $|$ and $-$ cannot be freely transformed into shapes with 1 bend.

5.1 Known B_1-VPG representations

Before we present our own results, we discuss some of the classes of graphs that were already known to have B_1-VPG representations.

5.1.1 Planar bipartite graphs

An example of a graph class that has a B_1-VPG contact representation are planar bipartite graphs. In this case, all the curves can be required to have no bends at all, so the representation is $\{|, -\}$-contact.

Lemma 5.1 (H. de Frasseix, P. O. de Mendez, J. Pach [38])**.** *A graph has a* $\{|, -\}$*-contact representation if and only if it is planar and bipartite.*

We will sketch this construction (and most other existing constructions for B_1-VPG representations), mostly so that the reader can get an idea just how varied and different the approaches are.

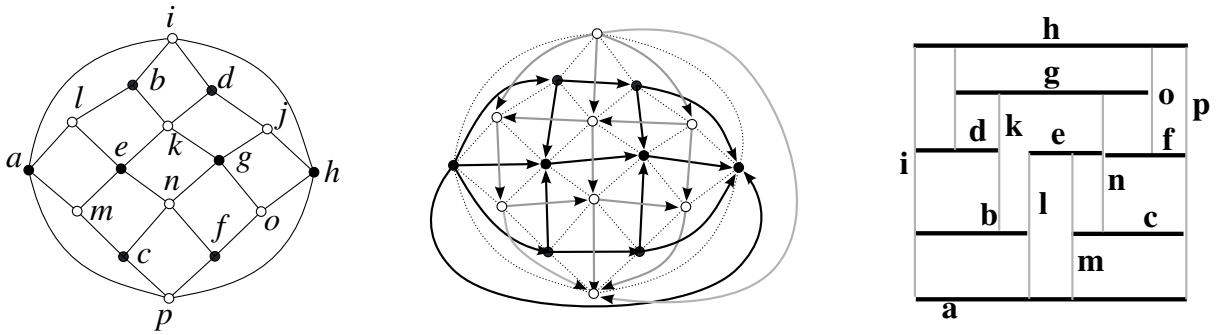


Figure 5.1: (a) An example of a bipartite graph. (b) Bipolar orientations. (c) Matching contact representation.

Proof sketch of Lemma 5.1. A *bipolar orientation* of a graph G is an orientation of edges which has (a) no oriented cycles; (b) two vertices s and t such that s has in-degree 0 and t has out-degree 0; and (c) $indeg(v) \geq 1$ and $outdeg(v) \geq 1$ for all $v \neq s, t$.

Let G be a plane bipartite graph. By adding vertices into faces of length more than 4, we can obtain a plane bipartite graph H that has all faces of length 4 and contains G as an induced subgraph. For any face bounded by vertices $b_1w_1b_2w_2$ in this order, let us add a black edge b_1, b_2 and a white edge w_1w_2, and denote the graphs induced by the black and white edges by H_B and H_W respectively. Let the outerface of H consist of vertices $v_1v_2v_3v_4$ so that v_1v_3 are black and b_2b_4 are white. One can show that H_B has a bipolar orientation from v_1 to v_3. This induces a dual orientation of H_W: For any interior face bounded by vertices $b_1w_1b_2w_2$ in clockwise order, $b_1 \to b_2 \Leftrightarrow w_1 \to w_2$.

Represent every vertex v in H_B by a horizontal segment so that the source $s = v_1$ has the y-coordinate $y(v_1) = 0$ and for every $b_i \to b_j$ we have $y(b_i) < y(b_j)$. Similarly, represent the vertices in H_W by vertical segments so that v_2 has the x-coordinate $x(v_2) = 0$ and for any $w_i \to w_j$ we have $x(w_i) < x(w_j)$. One can then prove that the lengths and positions of these segments can be set appropriately so that only the required contacts are made. See Figure 5.1 for an example. □

5.1.2 Series-parallel graphs

An example of a graph class with B_1-VPG-contact representation that require some bends is the class of the *series-parallel graphs* (see e.g. [23]). A *series-parallel* graph is a graph with a pair of designated distinct vertices (s, t) (so-called *terminals*) such that the graph is either an edge (s, t) or can be obtained by one of the two following constructions:

1. *Series composition:* Given a two series-parallel graphs with terminals (s_1, t_1) and (s_2, t_2), obtain a graph by identifying t_1 with s_2 and set terminals to (s_1, t_2).
2. *Parallel composition:* Given a two series-parallel graphs with terminals (s_1, t_1) and (s_2, t_2), obtain a graph by identifying s_1 with s_2 and t_1 with t_2, and set terminals to be $(s_1 = s_2, t_1 = t_2)$.

As we show in the following lemma, series-parallel graphs have B_1-VPG-contact representations, and in fact, only ⌊-shapes are required. We are not aware of a reference that states exactly this result, but it is implicit, e.g., in [34].

Lemma 5.2. *Every series-parallel graph has an* {⌊, ⁻, ⌈}*-contact representation enclosed in a rectangular area* Θ *such that*

- s *is represented by a horizontal segment* $\mathbf{s}$ *that lies on the top boundary of* Θ.
- t *is represented by a vertical segment* $\mathbf{t}$ *that lies on the right boundary of* Θ.
- *No other segment lies on the top or right boundary of* Θ.
- *Every* $v \neq s, t$ *is represented by the shape of an* ⌊.
- *There is a rectangle* Q *called the* private rectangle *in the top right corner of* Θ *situated on the top and right boundary of* Θ *and is intersected by* $\mathbf{s}$ *and* $\mathbf{t}$ *and no other curves. If* (s, t) *is not an edge, then neither* $\mathbf{s}$ *nor* $\mathbf{t}$ *contains the top right corner of* Q.

Proof. See Figure 5.2 for an illustration. Let G be a series-parallel graph. If G is an edge (s, t), it can be represented by a unit-length horizontal segment $\mathbf{s}$ meeting a unit-length

vertical segment $\mathbf{t}$ in the top right corner of Θ. The rectangle Θ is the rectangle Q of the representation.

Assume that two representations R_1 with curves $\mathbf{s_1}, \mathbf{t_1}$ and R_2 with curves $\mathbf{s_2}, \mathbf{t_2}$ are given. Let those representations be enclosed by rectangles Θ_1 and Θ_2 and have private rectangles Q_1 and Q_2, respectively.

For a series composition, place Θ_2 below and right of Θ_1. Extend $\mathbf{t_1}$ downwards and $\mathbf{s_2}$ leftwards until they meet. Note that the shape of this vertex becomes an $\llcorner$. The representation is enclosed in a rectangle Θ whose top boundary contains $\mathbf{s_1}$, and the right boundary contains $\mathbf{t_2}$. Note there is no edge (s_1, t_2) and there is an empty rectangle at the top right corner of Θ. After extending $\mathbf{s_1}$ rightward and $\mathbf{t_2}$ upward, we can find a suitable rectangle Q here. Note that extending $\mathbf{s_1}$ is feasible without adding new contacts since either $(s_1, t_1) \in E$ or $\mathbf{t_1}$ does not contain the top right corner of Q_1. Similarly one argues that t_2 can be extended.

For a parallel composition, if one of the graphs is just an edge, without loss of generality assume that it is (s_1, t_1), and extend $\mathbf{s_2}$ to meet $\mathbf{t_2}$ in Θ_2. Otherwise, shrink Θ_2 and place it into Q_1. Extend $\mathbf{s_2}$ leftwards until it meets $\mathbf{s_1}$ if needed, and extend $\mathbf{t_2}$ downwards until it meets $\mathbf{t_1}$ if needed. As before, one argues that this creates no unwanted contacts. The resulting representation is enclosed in Θ_1 and contains a private rectangle Q_2 as required. □

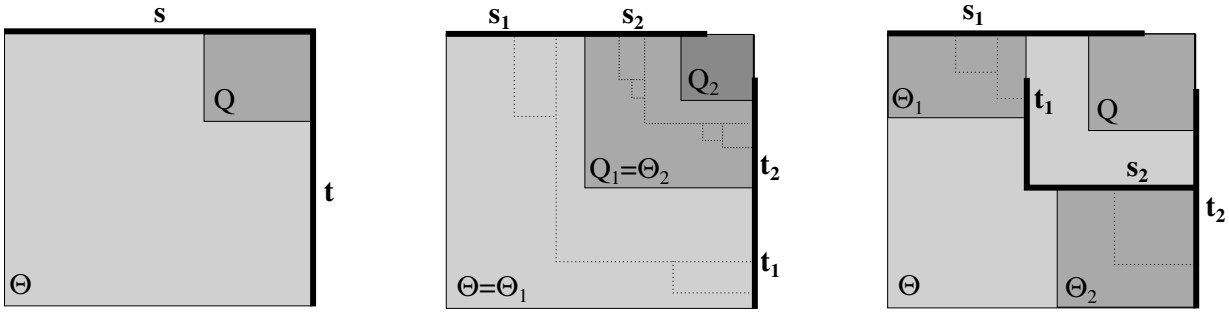
We can convert the curves of s and t into $\llcorner$'s as well by adding dummy segments at their ends, and hence obtain:

Corollary 5.3. *Every series-parallel graph has an $\{\llcorner\}$-contact representation.*

It is well-known that every outer-planar graph is a subgraph of a series-parallel graph. Thus, Corollary 5.3 implies that every outer-planar graph has a $\{\llcorner\}$-contact representation.

5.1.3 Laman graphs

A *Laman graph* is a connected graph $G = (V, E)$ with $|E| = 2|V| - 3$ such that for every induced subgraph $G[W]$ of G, we have $|E(G[W])| \leq 2|W| - 3$. It can be shown [65] that



(a) (b) (c)

Figure 5.2: Constructing a B_1-VPG-contact representation of a series-parallel graph. (a) The base case. (b) Parallel composition. (c) Series composition.

every Laman graph can be constructed from a triangle by a sequence of the following the *Henneberg constructions*:

(H1) Connect two vertices x, y with an edge e and subdivide e with a new vertex.

(H2) Choose an edge $e = (x, y)$ and a third vertex z, subdivide e with a vertex w and connect w to z.

If G is a planar graph, the operations can be performed so that all the intermediate graphs are planar as well.

Lemma 5.4 (Kobourov, Ueckerdt, Verbeek [65])**.** *If G is a planar Laman graph, then it has a $\{\llcorner, \ulcorner, \lrcorner, \urcorner\}$-contact representation.*

For the proof of Lemma 5.4, the authors showed that given the Henneberg construction of a planar embedded Laman graph G, one can compute a combinatorial structure called an *angular tree*. This structure can then be used to produce a labeling of angles and edges in an embedded planar Laman graph, and subsequently compute a $\{\llcorner, \ulcorner, \lrcorner, \urcorner\}$-contact representation. The construction runs in time $O(n^2)$ which is required for computing the angular tree (it can be performed in linear time if an angular tree is given). The details are quite complicated and will be omitted here.

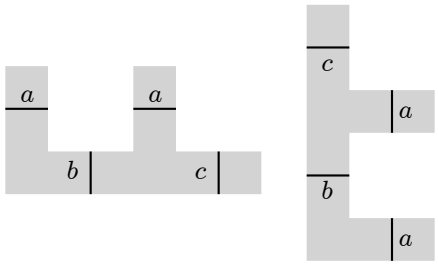


Figure 5.3: The two possible shapes of private regions from [42].

Note that Laman graphs also have contact representations using line segments [42, 6]. However, those segments do not need to be axis-aligned, thus those representations are not VPG representations. Other characterizations of B_1-VPG-contact graphs with connections to Schnyder realizers and canonical orders of maximally planar graphs were provided in [34].

5.1.4 Planar 3-trees

A 3-*tree* (see e.g. [23]) is either a triangle, or a graph that can be obtained from a 3-tree by inserting a new vertex of degree 3 and attaching it to a triangle.

Lemma 5.5 (Felsner et al. [42])**.** *All planar* 3*-trees have* $\{\llcorner\}$*-representations.*

We briefly review the construction proving Lemma 5.5 here as it serves as inspiration for our construction for planar *partial* 3-trees presented in Section 5.2.1.

The goal of the construction is to produce a special kind of an $\{\llcorner\}$-representation that satisfies the additional property that for every inner triangular face $\{a, b, c\}$, there exists a subset of the plane, called the *private region of the face*, that intersects only the curves **a**, **b** and **c**. Furthermore, there are only two possible shapes for the private regions (shown in Figure 5.3), there is a prescribed way that the curves intersect private regions, and the private regions of all faces are disjoint. (We used a similar concept in Chapter 3.) The representation can be built inductively, following insertions of vertices of degree 3 into faces, where curves are inserted into private regions as shown in Figure 5.4.

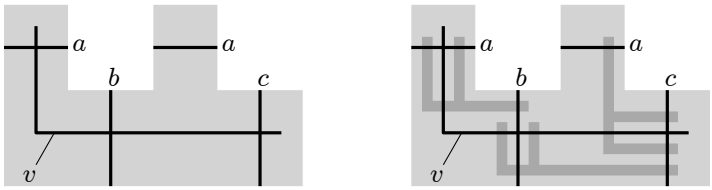


Figure 5.4: Building the representation by inserting a curve **v** and associated private regions into the private region for triangular face a, b, c [42].

Note that planar 3-trees are unrelated to planar Laman graphs (which have B_1-VPG-contact representation, see Lemma 5.4) because Laman graphs have $2n - 3$ edges while 3-trees have $3n - 6$ edges, so not every planar 3-tree is a Laman graph.

5.1.5 Other graph classes

There are several other results about B_1-VPG representations worth mentioning here. We provide only a list without any further details as our work does not use these results.

Given a graph $G = (V, E)$, its *line graph* H is obtained by using one vertex for every edge of G and adding an edge (e, e') whenever e and e' share an endpoint. Felsner et al. showed the following [42]:

Lemma 5.6. *Every line graph of a planar graph has an* $\{\llcorner\}$*-representation.*

The construction uses the so-called *canonical ordering* and builds the representation by incrementally following this order; see [42] for more details.

Middendorf and Pfeiffer [74] showed that the complement of any even subdivision of any graph, i.e., every edge is subdivided with a non-zero even number of vertices, has a B_1-VPG representation that uses only shapes in $\{\llcorner, \urcorner\}$.

A special kind of $\{\llcorner\}$-representations was investigated by Ahmed et al. [3] in 2017. Their work is concerned with $\{\llcorner\}$-representations where the corners of $\llcorner$'s lie on a lie segment in an xy-monotone fashion. Such a representation is called $\llcorner$-monotone. The authors provide a complete characterization of graphs with $\llcorner$-monotone representations as so-called

non-jumping graphs. Those include all outer-planar graphs, convex bipartite graphs and chordal graphs that exclude 3 forbidden subgraphs. See [3] for the definitions and more details. Furthermore, the authors show that not all graphs with {⌊}-representations have ⌊-monotone representations, and that not all planar graphs are ⌊-monotone. Some of these results turned out to be known earlier, see, e.g., [28, 32].

5.2 New B_1-VPG representations

In the following three sections, we present our new results on B_1-VPG representations. We first extend the result of [42] and show that all planar partial 3-trees have B_1-VPG representations. We then consider special subclasses of planar partial 3-trees, so-called IO-graphs and Halin graphs, and show that only a subset of shapes is necessary for their representations.

5.2.1 Planar partial 3-trees

Recall the definition of a 3-tree from Section 5.1.4: a 3-*tree* is a graph that is either a triangle or has a vertex order $v_1, \ldots, v_n$ such that for $i \geq 4$, vertex v_i is adjacent to exactly three predecessors and they form a triangle. A *partial* 3-*tree* is a subgraph of a 3-tree. Thus, partial 3-trees include all 3-trees. The class of partial 3-trees also has a non-empty intersection with planar Laman graph, but there is no inclusion relationship between the two.

Our construction in this section is similar to the construction of Felsner et al. [42] of Lemma 5.5 showing that every planar 3-tree (not partial) has an {⌊}-representation. Note that this implies that all 3-trees are 1-string as every {⌊}-representations is a 1-string representation. Naturally one wonders whether this carries over to planar partial 3-trees. Generally, the property of having a string representation is not closed under taking subgraphs. However, planar partial 3-trees inherit the recursive structure from their supergraphs. Thus, we consider it likely that the technique of [42] would work for planar partial 3-trees. We succeeded partially: We can find a B_1-VPG representation, but we need

to use all four possible shapes ⌞, ⌜, ⌝ and ⌟. Our construction can be performed in linear time.

In this section, we present the construction while proving the following theorem:

Theorem 5.7. *Every planar partial* 3*-tree* G *has a* 1*-string* B_1*-VPG representation.*

Our proof of Theorem 5.7 employs the method of "private regions" used previously for various string representation constructions [12, 31, 42] (and also in Chapter 3), but uses some different shapes. We define the following (see Figure 5.5):

Definition 5.8 (F-shape and rectangular shape)**.** *An* F-shaped area *is a region bounded by a* 10*-sided polygon with CW or CCW sequence of interior angles* 90°*,* 270°*,* 90°*,* 90°*,* 270°*,* 270°*,* 90°*,* 90°*,* 90° *and* 90°*. A* rectangle-shaped area *is a region bounded by an axis-aligned rectangle.*

Definition 5.9 (P3T-private region)**.** *Given a* 1*-string representation, a* P3T-private region[2] *of vertices* $\{a, b, c\}$ *is an F-shaped or rectangle-shaped area that intersects (up to permutation of names) curves* $\mathbf{a}, \mathbf{b}, \mathbf{c}$ *in the way depicted in Figure 5.5(a), and that intersects no other curves and P3T-private regions.*

Note that the F-shape P3T-private region was already used for planar 3-trees (see Section 5.1.4) while the rectangle-shaped region is a new concept.

Now we are ready to prove Theorem 5.7. Let G be a planar partial 3-tree. By definition, there exists a 3-tree H for which G is a subgraph. One can show [17] that we may assume H to be planar. Let $v_1, \ldots, v_n$ be a vertex order of H such that for $i \geq 4$ vertex v_i is adjacent to 3 predecessors that form a triangle. In particular, v_4 is incident to a triangle formed by $\{v_1, v_2, v_3\}$. One can show (see e.g. [17]) that the vertex order can be chosen in such a way that $\{v_1, v_2, v_3\}$ is the outer face of H in some planar drawing.

For $i \geq 3$, let G_i and H_i be the subgraphs of G (respectively H) induced by vertices $v_1, \ldots, v_i$. We prove Theorem 5.7 by showing the following by induction on i:

[2]The notion of private regions is used several times this thesis. In order to distinguish the types in various proofs, here we use "P3T" (as in "Partial 3-Tree") in the name of the private region.

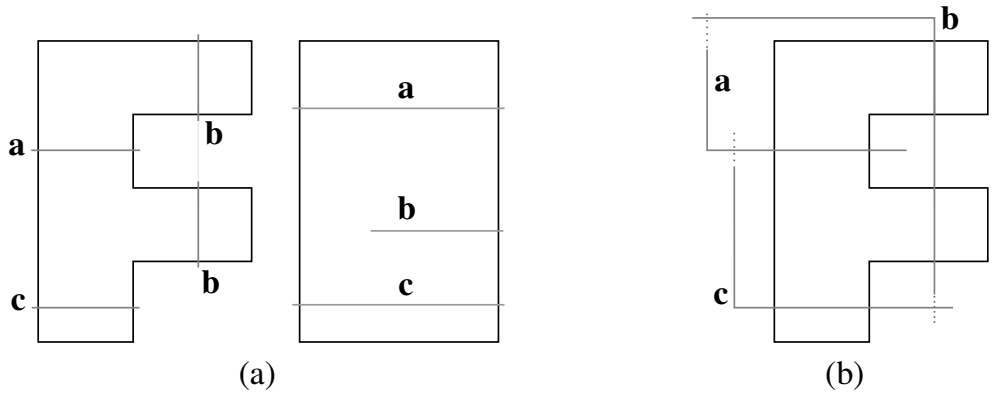


Figure 5.5: (a) An F-shaped (left) and rectangle-shaped (right) P3T-private region of $\{a, b, c\}$. (b) The base case of the construction of a B_1-VPG representation for a planar partial 3-tree.

> G_i has a 1-string B_1-VPG representation with a P3T-private region for every interior face of H_i.

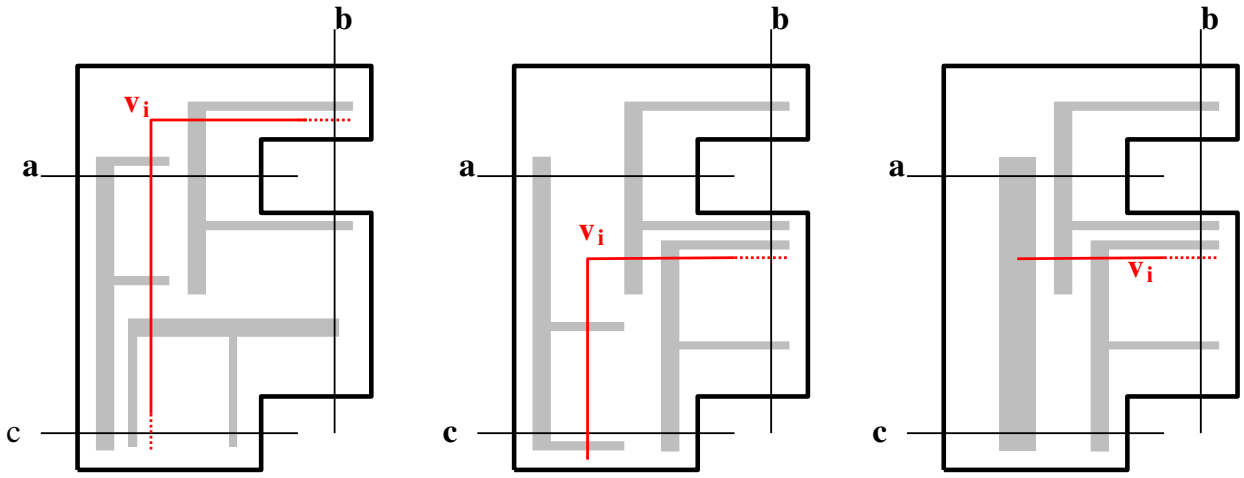
In the base case, $i = 3$ and $G \subseteq K_3 \simeq H$. Construct a representation R and find a P3T-private region for the unique interior face of H as depicted in Figure 5.5(b). Intersections among $\{\mathbf{a}, \mathbf{b}, \mathbf{c}\}$ can be omitted as needed.

Now consider $i \geq 4$. By induction, construct a representation R_0 of G_{i-1} that contains a P3T-private region for every interior face of H_{i-1}.

Let $\{a, b, c\}$ be the predecessors of v_i in H_i. Recall that they form a triangle. Since H_i is planar, this triangle must form a face in H_{i-1}. Since $\{v_1, v_2, v_3\}$ is the outer face of H_i (and hence also of H_{i-1}), the face into which v_i is added must be an interior face, so $\{a, b, c\}$ is an interior face in H_{i-1}. Let P_0 be the P3T-private region that exists for $\{a, b, c\}$ in R_0; it can have the shape of an F or a rectangle.

Observe that in G, vertex v_i may be adjacent to any possible subset of $\{a, b, c\}$. This gives 16 cases (two possible shapes, up to rotation and reflection, and 8 possible adjacencies).

In each case, the goal is to place a curve $\mathbf{v_i}$ inside P_0 such that it intersects exactly the curves of the neighbours of v_i in $\{a, b, c\}$ and no other curve. Furthermore, having placed



(a) $(v_i, a) \in E$ (b) $(v_i, a) \notin E$, but $(v_i, c) \in E$ (c) $(v_i, a), (v_i, c) \notin E$

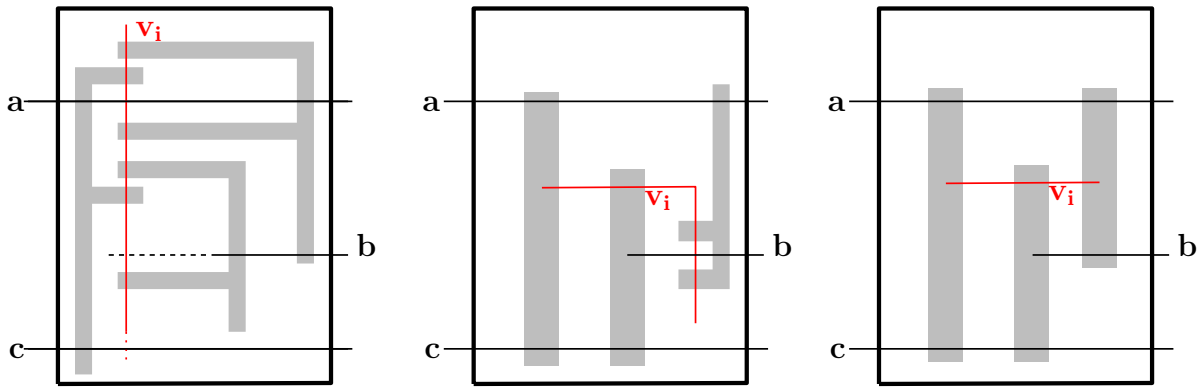
Figure 5.6: Inserting curve $\mathbf{v_i}$ into an F-shaped P3T-private region.

$\mathbf{v_i}$ into P_0, we need to find a P3T-private region for the three new interior faces in H_i, that is, the three faces formed by v_i and two of $\{a, b, c\}$.

Case 1: P_0 has the shape of an F.

After possible rotation / flip of R_0 and renaming of $\{a, b, c\}$ we may assume that P_0 appears as in Figure 5.5(a). If (v_i, a) is an edge, then place a bend for curve $\mathbf{v_i}$ in the region above **a** (Figure 5.6(a)). Let the vertical segment of $\mathbf{v_i}$ intersect **a** and (optionally) **c**. Let the horizontal segment of $\mathbf{v_i}$ intersect (optionally) the top occurrence of **b**. If (v_i, a) is not an edge but (v_i, c) is an edge, then place a bend for $\mathbf{v_i}$ in the region below **a** (Figure 5.6(b)), let the vertical segment of $\mathbf{v_i}$ intersect **c** and the horizontal segment of $\mathbf{v_i}$ intersect (optionally) **b**. Finally, if neither (v_i, a) nor (v_i, c) is an edge, then $\mathbf{v_i}$ is a horizontal segment in the region below **a** and above **c** (Figure 5.6(c)) that (optionally) intersects **b**.

In all sub-cases, $\mathbf{v_i}$ remains inside P_0, so it cannot intersect any other curve of R_0. Private regions for the newly created faces can be found as shown in Figure 5.6.



(a) (v_i, a) is an edge. (b) $(v_i, a), (v_i, c) \notin E$, $(v_i, b) \in E$. (c) $(v_i, a), (v_i, b), (v_i, c) \notin E$.

Figure 5.7: Inserting curve $\mathbf{v_i}$ and new private regions into a rectangle-shaped P3T-private region.

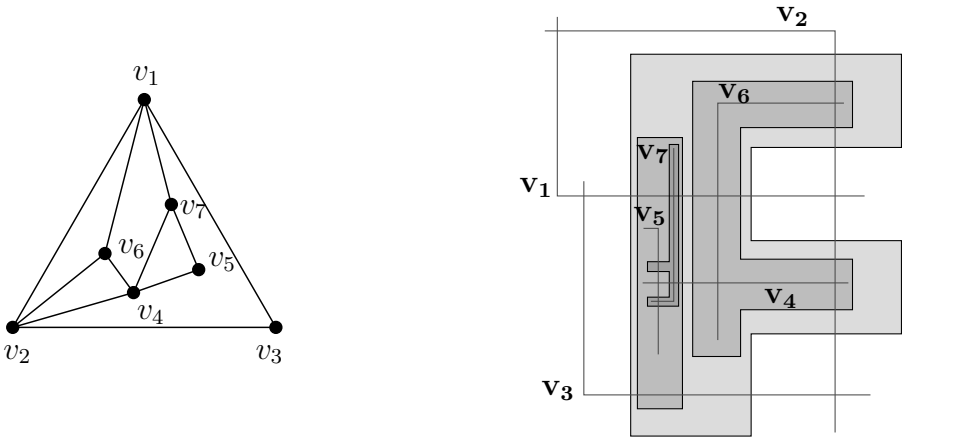
Case 2: P_0 has the shape of a rectangle.

After possible rotation / flip of R_0 and renaming of $\{a, b, c\}$ we may assume that P_0 appears as in Figure 5.7(a). If (v, a) is an edge, then $\mathbf{v}$ is a vertical segment that intersects $\mathbf{a}$ and (optionally) $\mathbf{b}$ and (optionally) $\mathbf{c}$. If (v, c) is an edge, then symmetrically $\mathbf{v}$ is a vertical segment that intersects $\mathbf{c}$ and (optionally) $\mathbf{b}$ and $\mathbf{a}$. Finally if neither (v, a) nor (v, c) is an edge, then let $\mathbf{v}$ be a horizontal segment between $\mathbf{a}$ and $\mathbf{c}$ with (optionally) a vertical segment attached to create an intersection with $\mathbf{b}$.

In all cases, $\mathbf{v}$ remains inside P_0, so it cannot intersect any other curve of R_0. Private regions for the newly created faces can be found as shown in Figure 5.7.

Theorem 5.7 now holds by induction. □

We note here that in our proof-approach, both types of P3T-private regions and all four shapes with one bend are required in some cases. An example of a graph that results in such a representation under a given elimination order is shown in Figure 5.8.



(a) A partial 3-tree with fixed elimination order.

(b) The representation constructed using P3T-private regions.

Figure 5.8: An example of a graph and elimination order that results in a representation with all four B_1-VPG shapes.

5.2.2 IO-Graphs

An *IO-graph* [41] is a 2-connected planar graph with a planar embedding such that the interior vertices form a (possibly empty) independent set. See Figure 5.10 for an example. IO-graphs are a subclass of planar partial 3-trees [41]. Here we show that, unlike for planar partial 3-trees, fewer shapes suffice in B_1-VPG representations of IO-graphs. We can show the following:

Theorem 5.10. *Any IO-graph has an* $\{\llcorner\}$*-representation.*

To prove Theorem 5.10, fix an IO-graph G. Let O be the set of exterior vertices; by definition these induce an outerplanar graph $G[O]$. Moreover, since G is 2-connected, the outer face is a simple cycle, and hence $G[O]$ is also 2-connected. We first construct an $\{\llcorner\}$-representation of $G[O]$, and then insert the interior vertices. To do so, we again use private regions, but we modify their definition slightly in three ways: (1) In an IO-graph, interior vertices may have arbitrarily high degree, and so the private regions must be allowed to cross arbitrarily many curves. (2) Interior vertices may only be adjacent to exterior

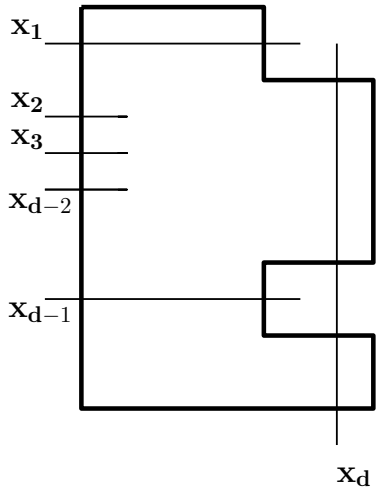


Figure 5.9: An IO-private region. We require that the supporting line of $\mathbf{x_i}$ (for $i = 2, \ldots, k-2$) intersects the upper segment of $\mathbf{x_d}$.

vertices. It therefore suffices for the private region of face f to intersect only those curves that belong to exterior vertices on f. It is exactly this latter observation that allows us to find private regions more easily, therefore use fewer shapes for them, and therefore use fewer shapes for the curves. We can therefore also add: (3) The private region must be an F-shape, and it must be in the rotation $\mathtt{E}$. The formal definition is given below:

Definition 5.11 (IO-private region)**.** *Given a 1-string representation of an IO-graph, an* IO-private region *of a face f is an F-shaped area P, in the rotation $\mathtt{E}$, that intersects curves $\mathbf{x_1}, \mathbf{x_2}, \ldots, \mathbf{x_d}$ as shown in Figure 5.9. Here, $\{x_1, \ldots, x_d\}$ is a subset of the vertices of f enumerated in CCW order, and includes all exterior vertices that belong to f (it may or may not include other vertices). Lastly, P intersects no other curves and no other private regions.*

Lemma 5.12. *Any outer planar graph has an $\{\llcorner\}$-representation with an IO-private region for every interior face.*

The representation can be obtained from Lemma 5.2 since every outerplanar graph is series-parallel. However, to argue the claim about private regions, we re-iterate the

construction here, specifically tailored to outerplanar graphs. This construction has been very recently re-discovered in [3].

Proof of Lemma 5.12. We may assume that the outerplanar graph is 2-connected, otherwise we can add vertices to make it so and delete their curves later. Enumerate the vertices on the outer face as $v_1, \ldots, v_k$ in CCW order. For every vertex v_i on the outerface, let $\mathbf{v_i}$ be an $\llcorner$ with the bend at $(i, -i)$. The vertical segment of $\mathbf{v_i}$ reaches until $(i, -r_i + \varepsilon)$, where $r_i = \min\{j : (v_j, v_i) \in E\}$. (Use $r_1 = 1$.) The horizontal segment of $\mathbf{v_i}$ reaches until $(s_i + \varepsilon, i)$, where $s_i = \max\{j : (v_j, v_i) \in E\}$. (Use $s_k = k$.) See also Figure 5.10.

It is quite easy to see that this is a 1-string representation. For every edge (v_i, v_k) with $i < k$ we have created an intersection at $(k, -i)$. Assume for contradiction that $\mathbf{v_i}$ and $\mathbf{v_k}$ intersect for some $(v_i, v_k) \notin E$ with $i < k$. Then we must have $s = \max\{j : (v_i, v_j) \in E\} > k$, else there is no intersection. Also $r = \min\{j : (v_j, v_k) \in E\} < i$, else there is no intersection. But then (v_i, v_s) and (v_j, v_r) are edges, and $\{v_i, v_j, v_s, v_r\}$ together with the outer face form a K_4-minor; this is impossible in an outer-planar graph.

Thus we found the $\{\llcorner\}$-representation. To find IO-private regions, we stretch horizontal segments of curves further as follows. For vertex v_i, set $t_i = \max\{j : v_i \text{ and } v_j \text{ are on a common interior face}\}$. If $t_i > s_i$, then expand $\mathbf{v_i}$ horizontally until x-coordinate $t_i - \varepsilon$. To see that this does not introduce new crossings, observe that adding (v_i, v_{t_i}) to the graph would not destroy outerplanarity, since the edge could be routed inside the common face. The $\{\llcorner\}$-representation of such an expanded graph would contain the constructed one and also contain the added segment. Therefore the added segment cannot intersect any other curves.

After stretching all curves horizontally in this way, an IO-private region for each interior face f can be inserted to the left of the vertical segment of $\mathbf{v_j}$, where v_j is the vertex on f with maximal index; see also Figure 5.10. □

Observe that for any crossing, one curve ends, so this representation can easily be converted into an $\{\llcorner\}$-contact representation (which is the one that Lemma 5.2 would have given).

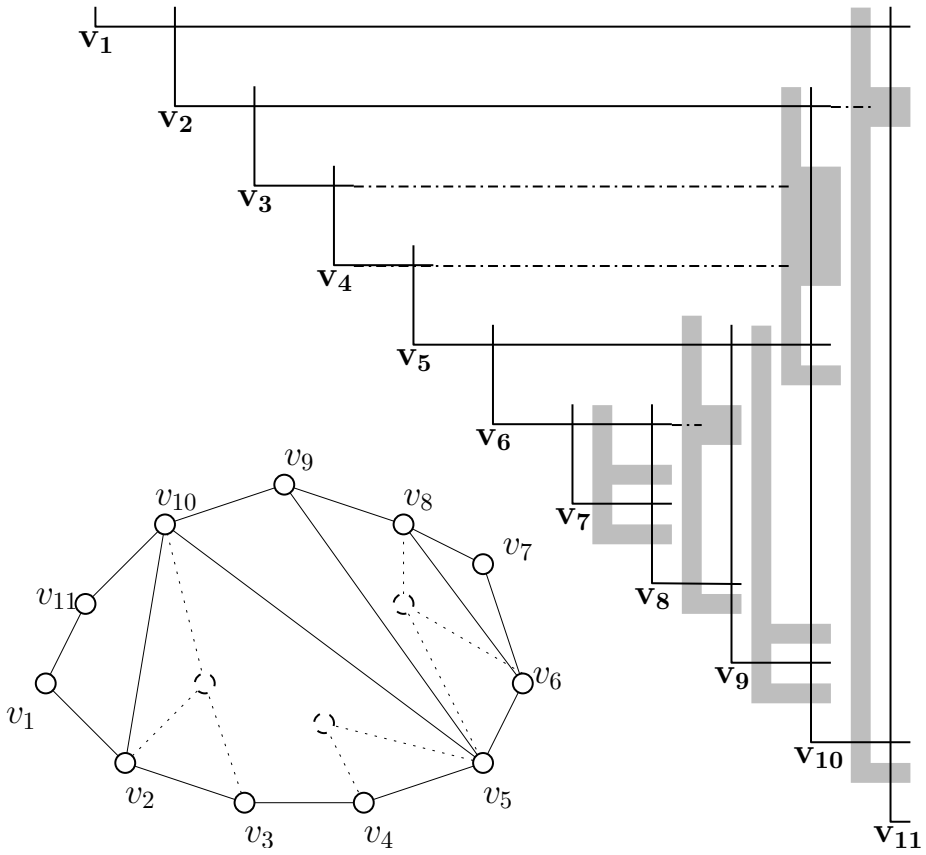


Figure 5.10: Example of an IO-graph and the $\{\llcorner\}$-representation of $G[O]$. The dotted vertices in the graph form an independent set that is attached to an outerplanar graph. The IO-private regions in the representations are shaded in grey.

Now we can prove Theorem 5.10, i.e., we can show that every IO-graph G has an $\{\llcorner\}$-representation. Start with the $\{\llcorner\}$-representation of $G[O]$ of Lemma 5.12. We add the interior vertices $v_1, \ldots, v_{n-k}$ to this in arbitrary order, maintaining the following invariant:

For every interior face of the current graph there exists an IO-private region.

Clearly this invariant holds for the representation of $G[O]$. Let v be the next interior vertex to be added, and let f be the face where it should be inserted. By induction there exists an IO-private region P_0 for face f such that the curves $\mathbf{x_1}, \ldots, \mathbf{x_d}$ that intersect P_0 include the curves of all exterior vertices that are on f, in CCW order. We need to place an $\llcorner$-curve $\mathbf{v}$ inside P_0, intersecting curves of neighbours of v and nothing else, and then find IO-private regions for every newly created face.

Since the interior vertices form an independent set, all neighbours of v are on the outer face, and hence belong to $\{x_1, \ldots, x_d\}$. Since G is 2-connected, v has at least two such neighbours. We have two cases (illustrated in Figure 5.11).

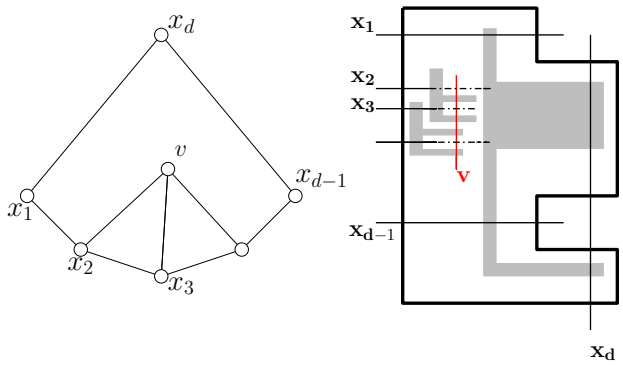
Case 1. If (v, x_d) is not an edge, then $\mathbf{v}$ is a vertical segment that extends from the topmost to the bottommost of the curves of its neighbours, and intersects these curves after expanding them rightwards.

Since the order of $\mathbf{x_1}, \ldots, \mathbf{x_d}$ is CCW around the outer face, for every newly created face f' incident to v we have a region inside P_0 in which the curves of outer face vertices on f' appear in CCW order. IO-private regions for these faces can be found as shown in Figure 5.11(top). Note that some of these private regions intersect v while others do not; both are acceptable since v is on those faces, but not an exterior vertex.

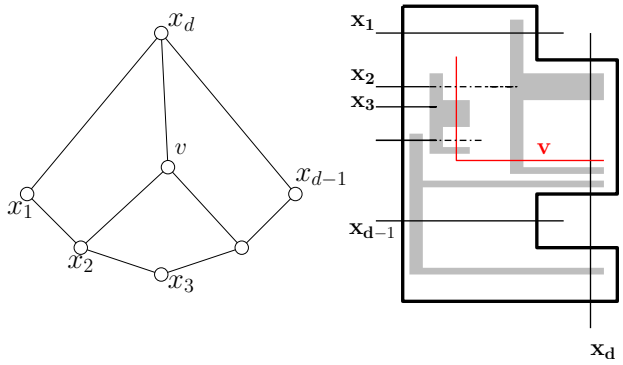
Case 2. If (v, x_d) is an edge, then $\mathbf{v}$ is an $\llcorner$, with the bend below $\mathbf{x_{d-1}}$ if (v, x_{d-1}) is an edge and above $\mathbf{x_{d-1}}$ otherwise. The vertical segment of $\mathbf{v}$ extends from this bend to the topmost of v's neighbours in $\{\mathbf{x_1}, \ldots, \mathbf{x_{d-1}}\}$, and intersects the curves of these neighbours after expanding them rightwards. The horizontal segment extends as to intersect $\mathbf{x_d}$.

IO-private regions can again be found easily, see Figure 5.11(middle and bottom).

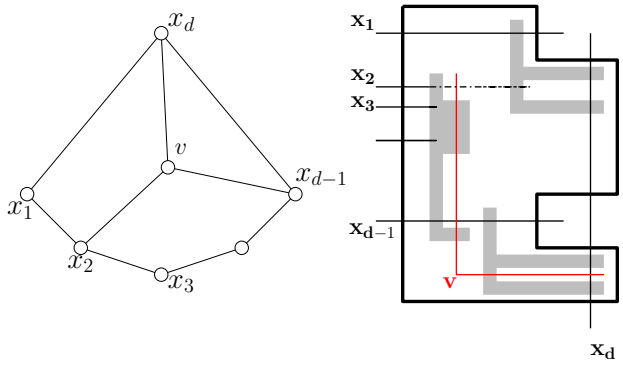
Repeating this insertion operation for all interior vertices hence gives the desired representation of G. □



(a) v is not adjacent to x_d.



(b) v is adjacent to x_d, but not x_{d-1}.



(c) v is adjacent to both x_d and x_{d-1}.

Figure 5.11: Inserting a vertex into a face of an IO-graph.

5.2.3 Halin graphs

A *Halin-graph* [57] is a graph obtained by taking a tree T with $n \geq 3$ vertices that has no vertex of degree 2 and connecting the leaves in a cycle. Such graphs were originally of interest since they are minimally 3-connected. Halin graphs are subgraphs of planar partial 3-trees [18].

Note that the existence of B_1-VPG representations for Halin graphs is easy to argue in two different ways. First, Halin graphs are partial 3-trees, so Theorem 5.7 applies. Second, they can be shown to be Laman graphs (unless they have only one interior vertex), and so Lemma 5.4 applies. However, both approaches potentially use all four B_1-VPG shapes and we hence give here a direct construction showing the following:

Theorem 5.13. *Any Halin-graph has:*

(a) a 1*-string* $\{\llcorner, \urcorner\}$*-representation, where only one vertex uses a curve of shape* $\urcorner$*.*

(b) an $\{\llcorner\}$*-representation.*

Note that an $\{\llcorner\}$-representation is 1-string too by definition. We prove both parts of Theorem 5.13 at once providing two very similar constructions. The significance of part (a) of Theorem 5.13 is that the very same construction can be easily modified to produce B_1-VPG-contact representations instead of B_1-VPG representations. Therefore, we will also obtain the following corollary:

Corollary 5.14. *Any Halin-graph has a* B_1*-VPG-contact representation such that every vertex is represented by a shape in* $\{\llcorner, \vert, \urcorner\}$*, only one vertex uses shape* $\urcorner$*, and one vertex uses shape* $\vert$*.*

Our construction will work even if the tree T has some vertices of degree 2. Fix an embedding of G such that the outer face is the cycle C connecting the leaves of tree T. Enumerate the outer face as $v_1, \ldots, v_k$ in CCW order. Since every exterior vertex was a leaf of T, vertex v_k has degree 3; let r be the interior vertex that is a neighbour of v_k. Root T at r and enumerate the vertices of T in post-order as $w_1, \ldots, w_n$, starting with the leaves (which are $v_1, \ldots, v_k$) and ending with r.

Let G_i be the graph induced by $w_1, \ldots, w_i$. Call vertex v_j *unfinished* in G_i if it has a neighbour in $G - G_i$. For $i = k, \ldots, n$, we create an $\{\llcorner\}$-representation of $G_i - (v_1, v_k)$ that satisfies the following:

> For any unfinished vertex v, curve $\mathbf{v}$ ends in a horizontal rightward ray, and the top-to-bottom order of these rays corresponds to the CW order of the unfinished vertices on the outer face while walking from v_1 to v_k.

The $\{\llcorner\}$-representation of $G_k - (v_1, v_k)$ (i.e., the path $v_1, \ldots, v_k$) is obtained easily by placing the bend for $\mathbf{v_i}$ at $(i, -i)$, giving the vertical segment length $1 + \varepsilon$ and leaving the horizontal segment as a ray as desired. To add vertex w_i for $i > k$, let $x_1, \ldots, x_d$ be its children in T; their curves have been placed already. Insert a vertical segment for $\mathbf{w_i}$ with x-coordinate i, and extending from just below the lowest curve of $\mathbf{x_1}, \ldots, \mathbf{x_d}$ to just above the highest. The rays of $\mathbf{x_1}, \ldots, \mathbf{x_d}$ end at x-coordinate $i + \varepsilon$, while $\mathbf{w_i}$ appends a horizontal ray at its lower endpoint.

Since adding w_i means that $x_1, \ldots, x_d$ are now finished (no vertex has two parents), the invariant holds. Continuing until $i = n$ yields an $\{\llcorner\}$-representation of $G - (v_1, v_k)$. It remains to add an intersection for edge (v_1, v_k). To do so, we change the shape of $\mathbf{v_1}$. Observe that its vertical segment was not used for any intersection, and that its horizontal segment can be expanded until $(n + 1, -1)$ without intersecting anything except its neighbours. After this expansion, we add a vertical segment going downward at its right end. Since v_k is a neighbour of r, curve $\mathbf{v_k}$ ended when $\mathbf{r}$ was added, i.e., at x-coordinate $n + \varepsilon$, and we can extend it until x-coordinate $n + 1 + \varepsilon$. Hence $\mathbf{v_1}$ and $\mathbf{v_k}$ can meet at $(n + 1, -k)$ if we change the shape of $\mathbf{v_1}$ to $\urcorner$. We have hence proved Theorem 5.13(a). □

By retracting curves so that they only touch, the representation becomes a B_1-VPG-contact representation that only uses shapes $\llcorner$ and $\urcorner$, and $|$ (for r). Since our construction for Halin-graphs produces contact representations, any contact (or crossing) can be omitted. Our result therefore holds not only for Halin graphs, but also for any subgraph of a Halin graph. This concludes the proof of Corollary 5.14.

Independently of our our work, Francis and Lahiri [50] proved that Halin-graphs are in fact $\{\llcorner\}$-intersection graphs, confirming our conjecture from [11]. Our construction of

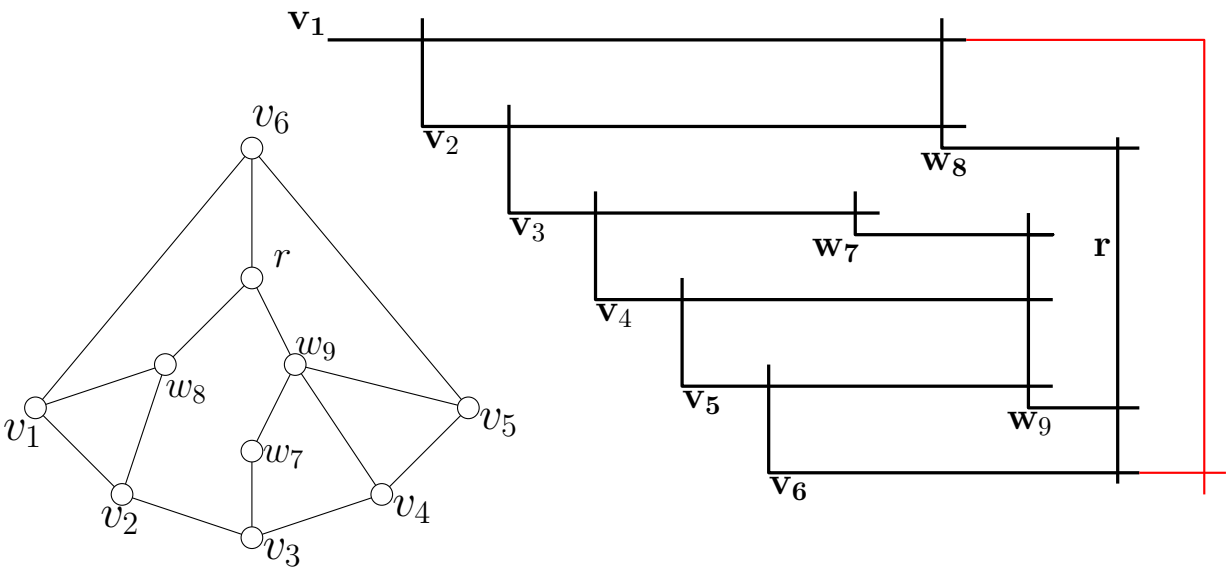


Figure 5.12: Example of an extended Halin-graph and its {⌊, ⌉}-representation, obtained by changing the curve of $\mathbf{v_1}$ so that it intersects $\mathbf{v_k}$.

contact representations presented above is able to prove this in the case where r has no neighbours on the outer face other than v_k. We can then change the vertex ordering so that it ends with the children of r in CCW order, followed by r and v_k. Then $\mathbf{r}$ can be a horizontal segment crossing the vertical segments of the children, and $\mathbf{v_k}$ can be placed entirely differently to intersect $\mathbf{v_{k-1}}, \mathbf{v_1}$ and $\mathbf{r}$. Figure 5.13a illustrates this idea.

If we do not require that the constructed representation is a contact representation, our construction can work with ⌊ shapes only as well. The main idea is as follows: The ⌉ shape is currently required to represent the edge (v_1, v_k). If r is adjacent to both v_{k-1} and v_k, we can represent $\mathbf{v_1}$ with an ⌊-shape instead, by not placing $\mathbf{v_k}$ in the initial step, but after placing $\mathbf{r}$. After placing $\mathbf{r}$, we can bend $\mathbf{r}$ rightwards and have $\mathbf{v_{k-1}}$ just above $\mathbf{r}$ in the bottom row, because it intersected $\mathbf{r}$, and $\mathbf{v_1}$ in the top row. Extend all three rightward and place $\mathbf{v_k}$ as vertical right-most segment there to hit them all. Figure 5.13b shows the obtained representation and the intersection between $\mathbf{v_{k-1}}$ and $\mathbf{r}$, which is now not a contact.

One can now argue that one of those two cases applies for a suitable choice of r and v_k:

either r has no neighbours on the outer face other than v_k, or r is adjacent to both v_k and v_{k-1}. This is can be achieved by choosing r to be the vertex where the two neighbours on the outer face are as close together as possible (the same argument is used in [50]). This finishes the proof of Theorem 5.13(b).

Finally, let us show that one cannot improve Theorem 5.13 to show that Halin graphs have {⌊}-contact representations:

Theorem 5.15. *Let G be a connected graph with $n \geq 2$ vertices that has an {⌊}-contact representation. Then G has at most $2n - 3$ edges.*

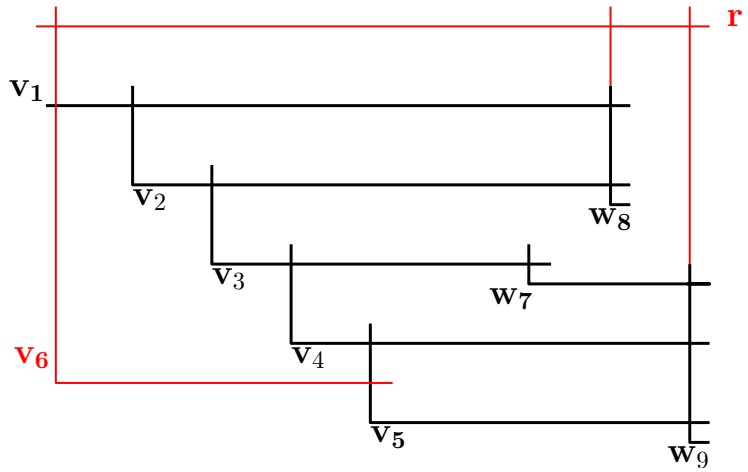
Proof. Any {⌊}-contact representation has two extreme ⌊-shapes:

- The *top-most* ⌊-shape with the maximum y-coordinate of the bend. Observe that the endpoint of its vertical segment does not make contact with any other ⌊-shape and can be extended to infinity upwards.
- The *right-most* ⌊-shape with the maximum x-coordinate of the bend. Observe that the endpoint of its horizontal segment does not make contact with any other ⌊-shape and can be extended to infinity rightwards.

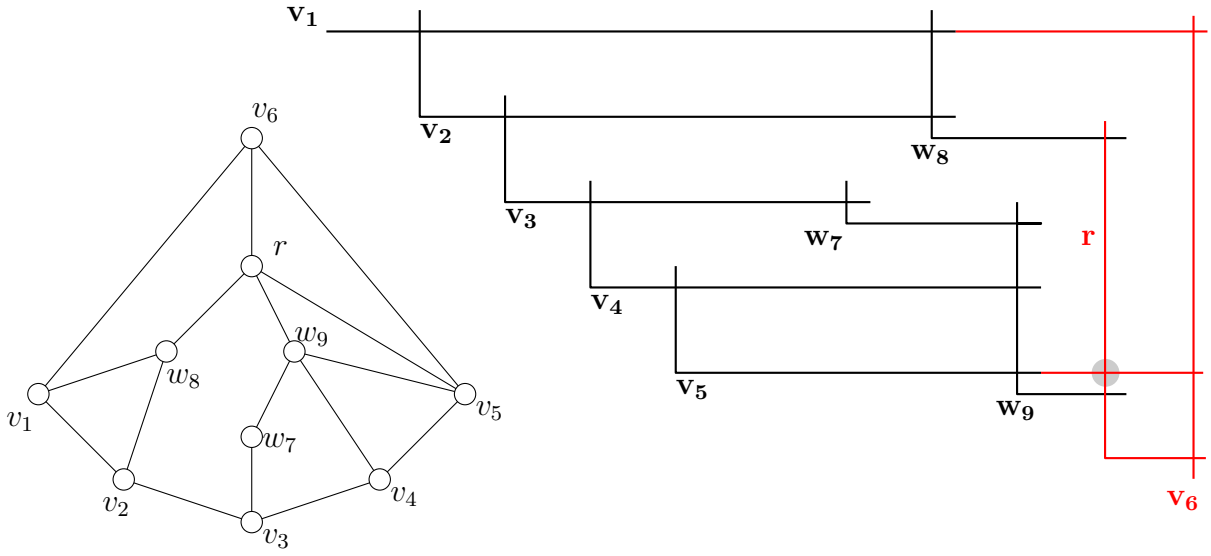
Note that the top-most and right-most ⌊-shapes are distinct unless the graph consists of a single vertex. We argue there is one more endpoint of an ⌊-shape that does not make a contact with any other shape. Consider the top-most ⌊. If its horizontal segment can be extended to infinity rightwards, we found the desired endpoint. Otherwise, it makes contact with another vertical segment of an ⌊-shape. Let us call that shape S; possibly S is the right-most ⌊-shape. Since the y-coordinate of the bend of the top-most ⌊ is maximum, the vertical segment of S cannot make a contact with any other ⌊.

As any edge in G corresponds to one endpoint contact, and there are at least 3 endpoints that do not make any contacts, G has at most $2n - 3$ edges. □

Note that since the top-most and right-most ⌊-shapes can be distinct, they can form a contact with each other, thus the bound of $2n - 3$ edges appears to be tight.



(a) Case 1: {⌊}-representation obtained by changing the curve of **r** and $\mathbf{v_k}$, if r has no other neighbours on the outer face. Refer to Figure 5.12 for the graph.



(b) Case 2: {⌊}-representation of a Halin graph with different placement of $\mathbf{v_k}$.

Figure 5.13: Constructing an {⌊}-representation of a Halin graph. The intersection that cannot be turned into a contact is highlighted.

A *wheel* W_k is a cycle C_k of length k together with a vertex v connected to all vertices of C_k. A wheel W_k is both a Halin and an IO-graph with $n = k + 1$ vertices and $2k = 2(k + 1) - 2 = 2n - 2$ edges. Thus, it is an example of a graph that cannot have an $\{\llcorner\}$-contact representation by Theorem 5.15.

Corollary 5.16. *There are IO-graphs and Halin graphs that do not have* $\{\llcorner\}$*-contact representations.*

5.3 Graphs with no B_1-VPG representations

In the previous sections, we provided some positive results and showed that some graph classes (planar partial 3-trees and all planar Laman graphs) have B_1-VPG representation. In this section, we present graphs that cannot have B_1-VPG representations.

There are two known constructions that can be used to produce graphs that do not have B_1-VPG representations. Chaplick et al. [33] showed that for any k, the class of graphs with B_{k+1}-VPG representations is strictly larger than the class of graphs with B_k-VPG representations. We present the graph obtained using their construction here. The other example is based on the construction of Kratochvíl and Matoušek from 1991 [70] and is presented in Chapter 7. Our hope had been to construct a planar graph that does not have a B_1-VPG representation, either directly from the construction, or by modifying it. We did not succeed [3], but the construction in Chapter 7 can be modified to give graphs that are close to planar in some sense.

Now we review the construction from [33]. Consider a closed rectangle whose boundary is formed by intersecting two orthogonal curves. There are two ways of creating such an area: one that requires two bends on one curve and none on the other curve; and one that requires a single bend on each curve. Thus, in order for two orthogonal curves to create two closed rectangular areas, at least one of them has to have at least two bends.

Let us fix one particular B_2-VPG representation R of $K_2 = (V = \{x, y\}, E = \{xy\})$ that gives rise to two closed rectangular areas A_1, A_2 so that each curve contributes precisely

[3] This is not surprising; see Section 8.2.

one bend to each of the areas (see the thick curves in Figure 5.15). Overlay a 5×5 grid over the representation so that every cell contains at most 1 intersection or endpoint of a curve of R. Call the constructed representation R'. Now, replace every grid vertex v with a gadget that consists of a horizontal *vertex segment* $S_1(v)$ and vertical vertex segment $S_2(v)$ that intersect each other. For every edge e between grid vertices u and v, replace the edge with three segments $S(u, e), S(e)$ and $S(v, e)$ so that (see Figure 5.14):

- the *edge segment* $S(e)$ has the same slope (horizontal or vertical) as e;
- the edge segment $S(e)$ intersects only the *edge connectors* $S(u, e)$ and $S(v, e)$;
- the edge connectors $S(u, e)$ and $S(v, e)$ have slopes opposite the edge e and intersect only $S(e)$ and $S_i(u)$ and $S_i(v)$, respectively, where $i \in \{1, 2\}$ is such that the slope of e and the intersected vertex segment are the same.

Call the final representation R''. Observe that the graph G'' obtained by replacing every intersection and bend of R'' with a vertex is a subdivision of a 3-connected planar graph, and as such, it has unique embedding (up to the choice of the outer face). Also, observe that the graph contains two separating cycles each formed by the boundary of the original rectangular areas A_1, A_2. Thus, any embedding of G'' has to contain two closed areas with disjoint interiors that correspond to A_1, A_2 that are bounded by subcurves of $\mathbf{x}$ and $\mathbf{y}$. Therefore, $\mathbf{x}$ and $\mathbf{y}$ together have at least 4 bends, and G'' has no B_1-VPG representation. The modified representation of G'' is shown in Figure 5.15. This graph is not planar (see Figure 5.15). It is close to planar in the sense that removing the two vertices (x and y) would make it planar, but there appears to be no way to modify the construction to make it planar.

5.4 Conclusions

In this chapter, we showed that every planar partial 3-tree has a 1-string B_1-VPG representation. We also showed that IO-graphs and Halin graphs have $\{\llcorner\}$-representations.

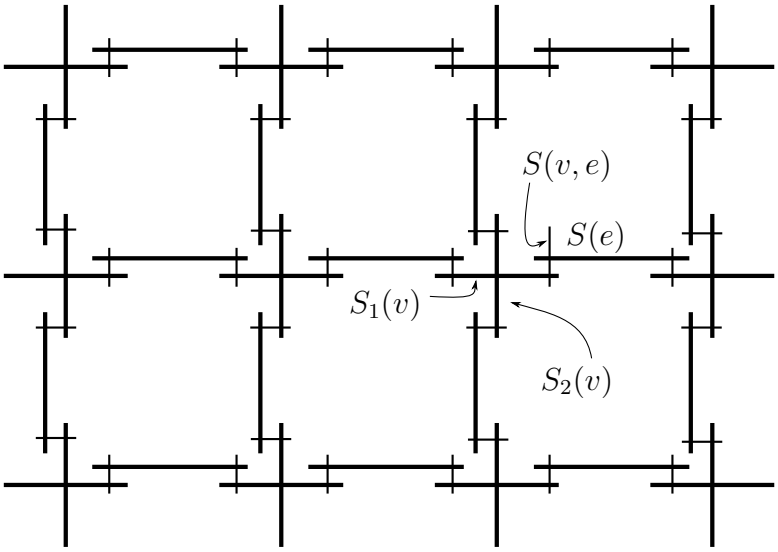


Figure 5.14: Replacing grid vertices with segment gadgets.

We also recalled the construction from [33] to show that some graphs are not B_1-VPG graphs. We presented one specific example of such a graph. Note that the graph is not 1-string (for the same reason that it is not B_1-VPG). This naturally raises the following question: is there a graph with a 1-string representation that is not B_1-VPG?

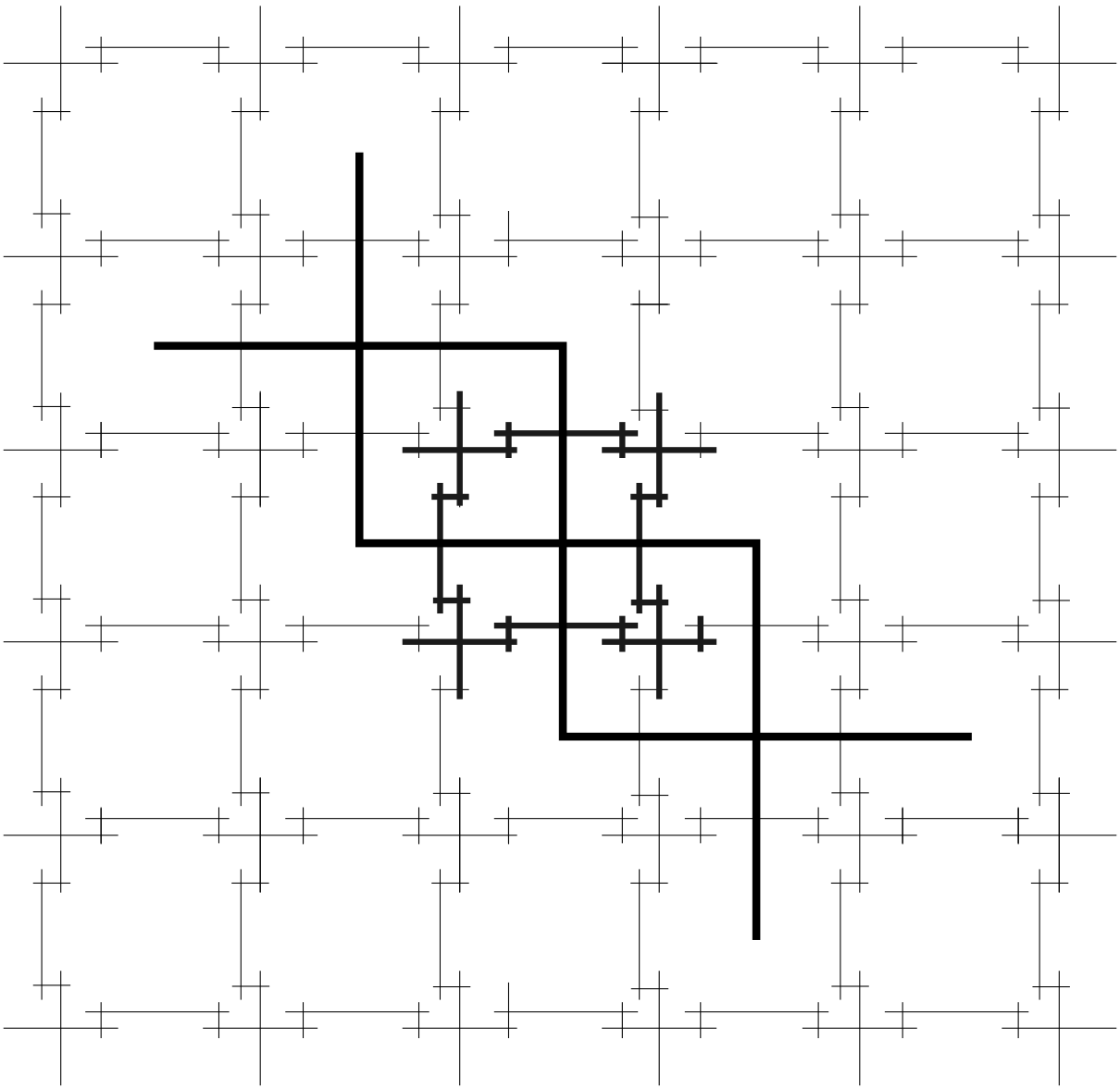


Figure 5.15: A B_2-VPG representation of a graph that does not have a B_1-VPG representation. Note that it is not planar: the two thick and the highlighted curves represent a subdivision of $K_{3,3}$.

Chapter 6

Order-Preserving String Representations

It is our experience that string representations are often hard to read, because the crossings of curves for edges occur at unexpected places. Thus, verifying if a string representation corresponds to a given graph with the naked eye is sometimes difficult.

Therefore, in this chapter, we study the following question: Does every planar graph have a 1-string representation where the order of crossings along curves *preserves* the planar embedding in the sense that the order of crossings along the curve of v corresponds to the cyclic order of edges around v in some planar embedding?

In addition to the aforementioned motivation, being able to show that planar graphs have order-preserving 1-string representations could make constructing such representations easier by using the typical incremental approach that adds one vertex on the outer-face at a time (recall that no short proof that planar graphs are in 1-String is known). For this it would be especially helpful if such representations were also *outer-string*. We show the following:

- We first discuss three possible variants of order-preservation. For the first variant (linear order-preserving), we present an algorithm that tests whether such a string-representation exists in linear time by reducing it to planarity testing.

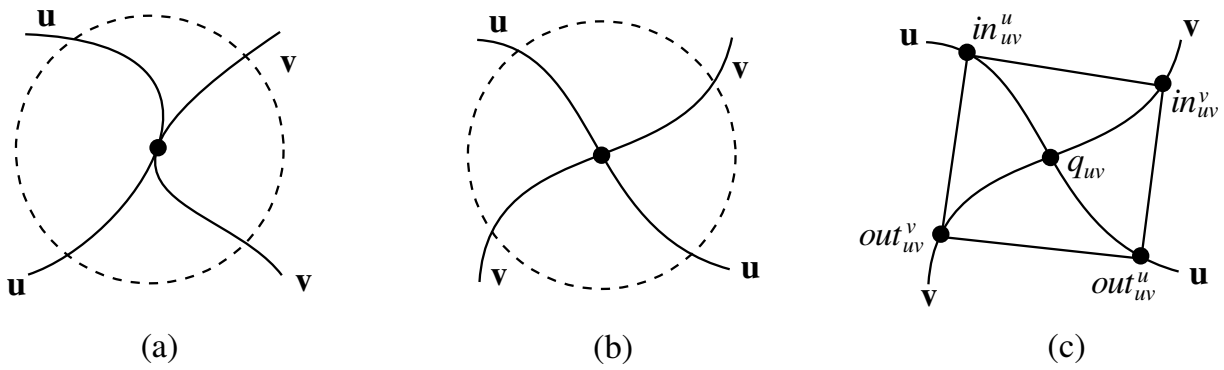
- For the second variant (cyclically order-preserving), we show that not all planar graphs have such a 1-string representations. In fact, we can construct a planar 3-tree that has no such representation.
- For some subclasses of planar partial 3-trees, we construct cyclically order-preserving 1-string representations. For outer-planar graphs, these are additionally outer-string (and use segments), while for the other graph classes we show that order-preserving outer-1-string representations do not always exist.
- The third variant (selectively order-preserving) defines the concept for string representations where two curves may intersect more than once. This variant also makes contact representations order-preserving with respect to planar embeddings that they imply.

Some results from this chapter were published in [14].

6.1 Linearly order-preserving 1-string representations

Let G be a planar graph given together with a combinatorial embedding, and for each vertex v of G let $L(v)$ be a list of v's neighbours in order as they appear around v in the embedding of G (breaking the cyclic order arbitrarily; this is a combinatorial embedding, except that we also fix who is first). Let R be a 1-string representation of G. We say that the order of crossings along curves in R *linearly preserves* (G, L) if the order of crossings along every curve $\mathbf{v}$ corresponds to $L(v)$. We also say that R is *linearly order-preserving* with respect to (G, L). Note that this model is geared specifically towards 1-string representations and linear order-preservation is undefined for representations that are not 1-string. Also note that the concept of linearly order-preserving string representations could be applied to any graph (possibly non-planar), as long as the lists $L(v)$ are determined in some way.

Lemma 6.1. *Given a graph G along with an ordered list of neighbours $L(v)$ for every vertex $v \in V(G)$, there is linear-time algorithm that decides whether G has a 1-string representation R that linearly preserves (G, L).*



(a) (b) (c)

Figure 6.1: (a) A realization of a crossing-vertex that does not induce a string representation. (b) A realization of a crossing-vertex that does induce a string representation. (c) A gadget that forces proper crossing-vertex realization.

Proof. We first sketch an idea that does not quite work. Let R be any 1-string representation of G. Let us construct a graph Q from R by replacing the crossings in R by dummy vertices. The representation R is then a drawing of Q in the plane, so Q is planar. Note that Q can be constructed directly from G and the ordered lists $L(v)$. One might think that given G and the order lists $L(v)$, one only needs to construct Q and test whether it is planar. However, the fact that Q is realizable as a planar graph does not imply the existence of a 1-string representation for G. The realization of Q can make two curves "touch" instead of making them intersect (see Figure 6.1(a)). To force an intersection, a proper order of edges around each dummy vertex is needed (see Figure 6.1(b)). Thus, instead of replacing crossings with vertices, we replace them with gadgets that force proper crossings and do not admit touches (see Figure 6.1(c)).

So, given a graph G and a list $L(v)$ for each vertex, create a graph H_0 as follows. For each edge (u, v) in G, add a vertex q_{uv}. These vertices represent the intersections. Then add four more vertices $in^v_{uv}, in^u_{uv}, out^v_{uv}, out^u_{uv}$ and connect them into a cycle (in this order), and connect them to q_{uv}. Observe that now in H_0, every vertex q_{uv} is part of the desired gadget. Finally, if $L(v) = \{u_1, u_2, \ldots, u_d\}$ then for $i = 1, \ldots, d-1$ connect $out^v_{u_i v}$ with $in^v_{u_{i+1} v}$.

We claim that H_0 is planar if and only if G has a linearly order-preserving 1-string repre-

sentation R. Given such an R, we replace the crossings with vertices q_{uv} and embed the remaining vertices on the curves connecting them. We can add the cycle $in^v_{uv}, in^u_{uv}, out^v_{uv}, out^u_{uv}$ in the vicinity of the crossing, since curves $\mathbf{u}$ and $\mathbf{v}$ cross properly.

Now assume that H_0 is planar, and fix a planar drawing Γ. For each $v \in V(G)$, let $\mathbf{v}$ be the curve consisting of edges induced by $\{in^v_{uv}, q_{uv}, out^v_{uv} | u \text{ intersects } v\}$ in Γ. This is a curve since we connected these vertices following the order of $L(v)$. Clearly, $\mathbf{v}$ and $\mathbf{u}$ have the point at q_{uv} in common. Since the gadget around q_{uv} is 3-connected, the edges at q_{uv} alternate between $\mathbf{u}$ and $\mathbf{v}$ and form a proper crossing. Due to our method of connecting edges, the order of intersections along $\mathbf{v}$ is exactly $L(v)$. Finally, no other intersections can exist since Γ is planar.

As planarity can be tested in linear time, this concludes the proof. □

It is easy to see that not all planar graphs have a linearly order-preserving 1-string representation; we show a stronger result in Theorem 6.3.

6.2 Cyclically order-preserving 1-string representations

Fix a combinatorial embedding of a graph. We say that a 1-string representation is *cyclically order-preserving* with respect to the combinatorial embedding if for any vertex v, we can walk along curve $\mathbf{v}$ from one end to the other and encounter the crossings with $\mathbf{w_1}, \ldots, \mathbf{w_k}$ in the same order in which the neighbours $w_1, \ldots, w_k$ of v appear in the cyclic order of edges around v. This leaves open the choice of which neighbour of v should be w_1, since the order at v is cyclic while the order along $\mathbf{v}$ is not. Throughout this section, we often write "order-preserving" when we mean "cyclically order-preserving."

6.2.1 Graphs with no cyclically order-preserving representations

In this section, we show that there exist planar graphs that have no cyclically order-preserving 1-string representation. To define them, we need again the stellation operation defined earlier. Recall that, given a plane graph G, the *stellation* of G is obtained by

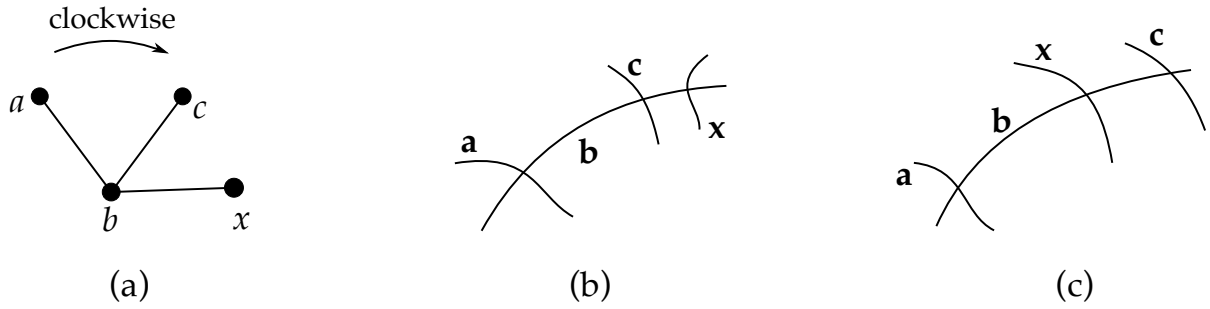


Figure 6.2: An illustration of incidences for the proof of Lemma 6.2. (a) A vertex with fixed rotation of the neighbours. (b) An unbroken incidence between **a** and **c**. (c) A broken incidence between **a** and **c**.

inserting a new vertex into every face of G, and making it adjacent to all vertices incident to that face. The *triple-stellation* of G is obtained by stellating G to get G', stellating G' to get G'', and finally stellating G''.

Lemma 6.2. *Let G be a plane graph with minimum degree 3 and at least $|V(G)|+1$ faces that are triangles. Then the triple-stellation G''' of G has no cyclically order-preserving 1-string representation with respect to this combinatorial embedding.*

Proof. Assume for contradiction we had such a 1-string representation $\mathcal{R}$, and let $\mathcal{R}_G$ be the induced 1-string representation of G, which is also order-preserving. The following notation will be helpful: If a, c are neighbours of b, then let $\mathbf{b}[a, c]$ be the stretch of **b** between the intersection with **a** and **c**.

Consider a face-vertex-incidence in G, which can be described by giving a vertex b and two neighbours a, c of b that are consecutive in the clockwise order at b. We call such a face-vertex-incidence *unbroken* if (in $\mathcal{R}_G$) $\mathbf{b}[a, c]$ contains no other crossing, else we call it *broken* (see Figure 6.2). Since $\mathcal{R}_G$ is order-preserving, for every vertex b in G only one face-vertex-incidence at b is broken[1]. Since G has at least $|V(G)|+1$ triangular faces, there exists a face $T = \{u, v, w\}$ of G such that all face-vertex-incidences at T are unbroken.

[1]Using this terminology, the only difference between linear and cyclic order preservation is that the former prescribes which face-vertex incidence is broken, while the latter leaves this choice free.

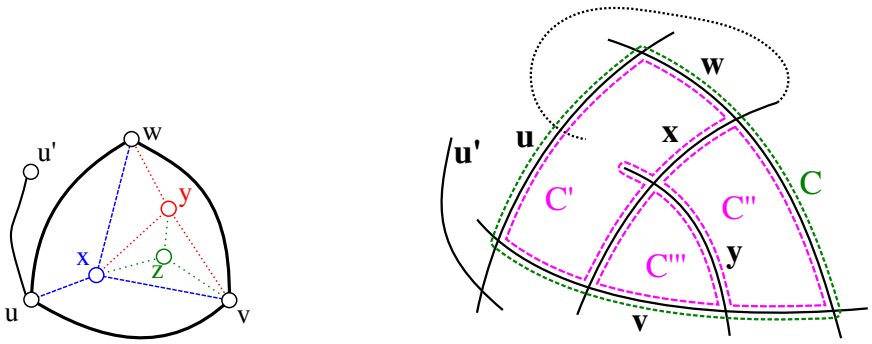


Figure 6.3: For the proof of Lemma 6.2.

Thus $\mathbf{u}[v,w]$, $\mathbf{w}[u,v]$ and $\mathbf{v}[u,w]$ all contain no other crossing in R_G, and hence bound a finite region C in R_G. We will find a contradiction at the stellation vertices that were placed in T and hence must intersect with C in special ways. See also Figure 6.3.

Let x be the vertex that (during the stellation of G to get G') was placed in face T. We claim that $\mathbf{x}$ must intersect $\mathbf{u}$ in $\mathbf{u}[v,w]$. To see this, recall that $\deg_G(u) \geq 3$, hence u has at least one other neighbour u' in G. Since the face-incidence at u is unbroken, $\mathbf{u}[v,w]$ contains no other crossing of $\mathcal{R}_G$, so $\mathbf{u}'$ intersects $\mathbf{u}$ outside this stretch. Since T is a face in G, the (clockwise or counter-clockwise) order of neighbours at u in G' contains u', v, x, w. To maintain this order in the string representation, the intersection between $\mathbf{x}$ and $\mathbf{u}$ (in $\mathcal{R}$) must be on $\mathbf{u}[v,w]$. Similarly one argues that $\mathbf{x}$ intersects $\mathbf{v}[u,w]$ and $\mathbf{w}[u,v]$.

Recall that C is the region bounded by $\mathbf{u}[v,w] \cup \mathbf{w}[u,v] \cup \mathbf{v}[w,u]$; this is a face of R_G, but may get partitioned by vertices inserted when stellating G. Curve $\mathbf{x}$ intersects δC three times by the above, and no more since curves intersect at most once in a 1-string representation. So $\mathbf{x}$ starts (say) inside C, crosses δC to go outside, crosses δC to go inside, and then crosses δC again to end outside. Between the second and third crossing, $\mathbf{x}$ contains a stretch that is inside C; after possible renaming of $\{u,v,w\}$ we assume that this is $\mathbf{x}[v,w]$. This stretch splits C into two parts, say C' (incident to parts of $\mathbf{u}$) and C^r (incident to the crossing of $\mathbf{v}$ and $\mathbf{w}$).

Let y be the vertex that (during the stellation of G' to get G'') was placed in the face $\{v,w,x\}$ of G'. Since v, w, x all have degree 3 or more in G', as before one argues that $\mathbf{y}$ must intersect $\mathbf{x}[v,w]$, $\mathbf{w}[x,v]$ and $\mathbf{v}[w,x]$. Curve $\mathbf{y}$ intersects $\delta C'$ (in $\mathbf{x}[v,w]$), but cannot

intersect $\delta C'$ a second time, else it would cross $\mathbf{u}$ (but $(u, y) \notin E$) or would cross one of $\mathbf{x}, \mathbf{v}, \mathbf{w}$ twice (which is not allowed). Hence $\mathbf{y}$ starts inside C', then crosses $\mathbf{x}$, and then crosses one of $\mathbf{v}$ and $\mathbf{w}$. Up to renaming of $\{v, w\}$ we may assume that $\mathbf{y}$ crosses $\mathbf{v}$ first. Hence $\mathbf{y}[x, v]$ splits C^r into two parts, say C'' (incident to parts of $\mathbf{w}$) and C''' (incident to the crossing of $\mathbf{v}$ and $\mathbf{x}$).

Now finally consider the vertex z that was placed in $\{x, y, v\}$ when stellating G'' to obtain G'''. As before one argues that $\mathbf{z}$ has an end inside C', because it crosses $\mathbf{x}$ in stretch $\mathbf{x}[v, y] \subset \mathbf{x}[v, w]$, and it cannot cross C' again. But we can also see that $\mathbf{z}$ has an end inside C'', since it crosses $\mathbf{y}[x, v]$ and crosses no other curve on the boundary of C''. But this means that $\mathbf{z}$ has both ends outside C''', contradicting that it must intersect the boundary of C''' three times to respect the edge-orders at x, y, v. Contradiction, so G''' does not have an order-preserving 1-string representation. □

Theorem 6.3. *There exists a planar 3-tree that has no cyclically order-preserving 1-string representation.*

Proof. Start with an arbitrary planar 3-tree G with $n \geq 6$ vertices; this has minimum degree 3 and $2n - 4 \geq n + 2$ triangular faces in its (unique) combinatorial embedding. Stellating a 3-tree gives again a 3-tree, so by Lemma 6.2 the triple-stellation of G is a 3-tree that has no order-preserving 1-string representation. □

The smallest graph (see Figure 6.4) without order-preserving string representation derived from Theorem 6.3 has $6 + 8 \cdot 13 = 110$ vertices. We do not believe that it is the smallest example of a graph with no cyclically order-preserving embedding, but we do not know of a smaller one. Also, this graph is 3-outer-planar, i.e., if we remove the vertices from the outerface, and repeat the operation twice on the resulting graph, then all the vertices are removed (see Figure 6.4). Is there a 2-outer-planar example, i.e., a graph that can be entirely eliminated by removing the outerface vertices twice only? (We will see in Section 6.3 that outer-planar graphs have order-preserving 1-string representations.)

Corollary 6.4. *The algorithms for constructing B_1-VPG representations of (partial) 3-trees in Lemma 5.5 and Theorem 5.7 do not create cyclically order-preserving representations.*

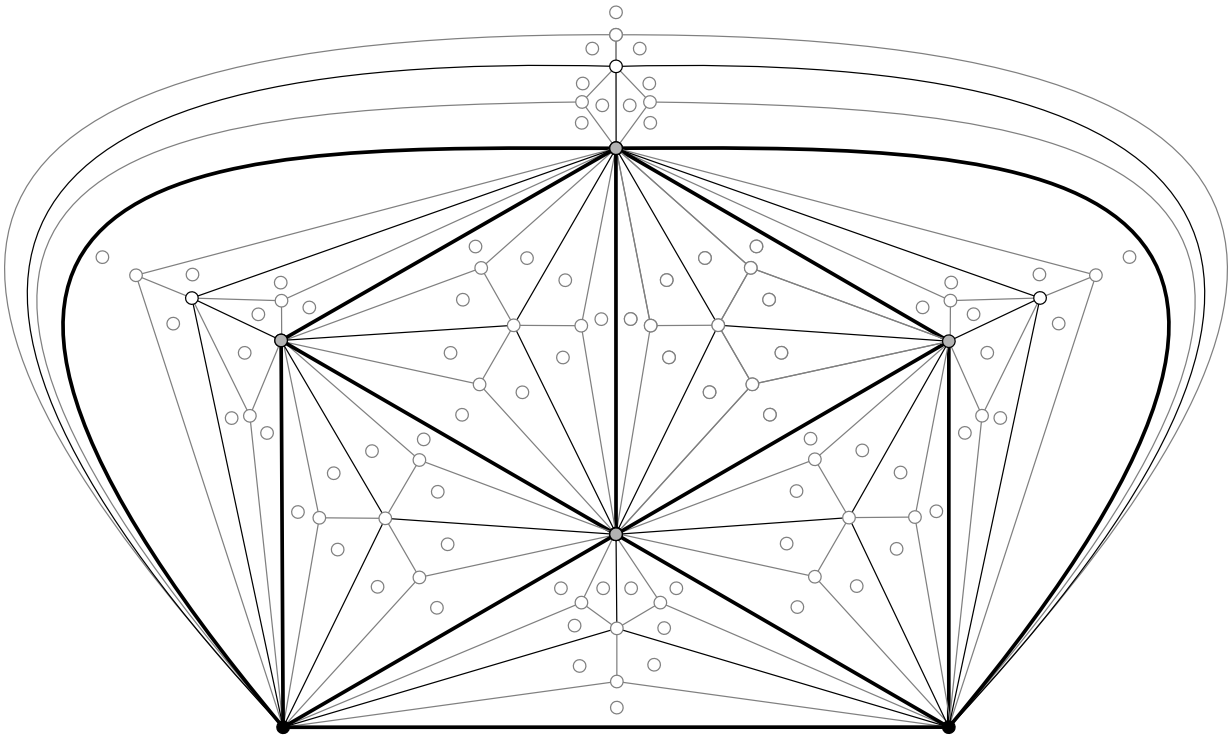


Figure 6.4: A triple stellation of a 3-tree with 6 vertices. The edges of the last stellation are not shown. The graph is 3-outer-planar—after removing the black outer-face vertices, the gray vertices are on the outer-face. Every other vertex has a black or gray neighbour.

Proof. These algorithms cover 3-trees and partial 3-trees which include 3-trees. The claim is immediately implied by Theorem 6.3. □

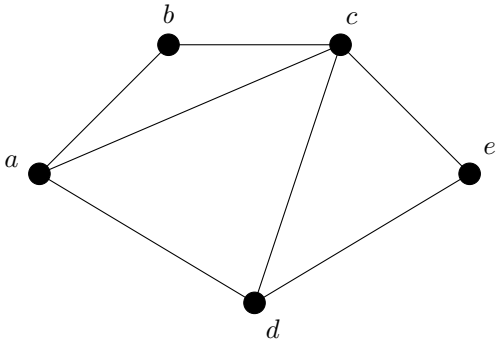
6.3 Outer-planar graphs

Now we turn towards positive results and show that every outer-plane graph has an order-preserving outer-1-string representation. We first discuss one existing result that does not quite achieve this. It is easy to show that every outer-planar graph can be represented as contact graph of line segments because it is a Laman graph [42] (see also Section 5.1.3). The standard way to do this (see also Figure 6.5) results, after extending the segments a bit, in a segment-representation that is order-preserving and such that every segment has a point visible from infinity. However, this does not quite achieve our goal, because the ends of segments are not necessarily on the outer-face which the definition of outerstring representations demands. We could bring an end of each segment to the outerface by tracing around segments, but then the representation would not be 1-string.

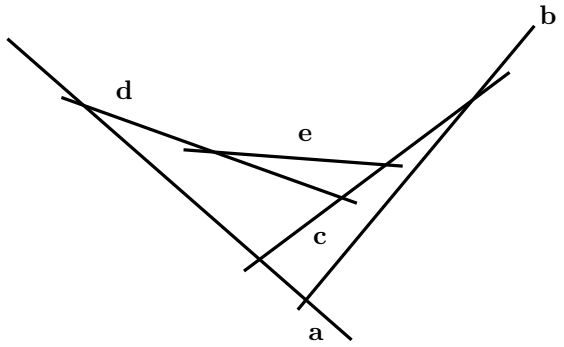
We instead give two other constructions. The first one uses the fact that any outer-planar graph is a *circle graph*, i.e., the intersection graph of chords of a circle [84]. This obviously gives an outer-segment representation, but it need not be order-preserving (see Figure 6.5). Our first construction hence re-proves this result and maintains invariants to ensure that the representation is indeed order-preserving.

The resolution in this representation could be very bad, and we therefore give a second construction where the curves are orthogonal instead. We use one bend for each vertex curve here, and so obtain a B_1-VPG-representation. Since there are n vertices and at most n bends, the representation can be embedded into a grid of size $O(n) \times O(n)$.

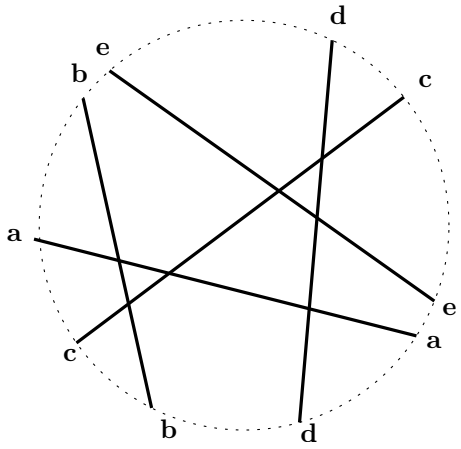
In our proofs, we use that any 2-connected outer-planar graph G can be built up as follows [56, Lemma 3]: Fix an edge (u, v). Now repeatedly add an *ear*, i.e., a path $P = u_0, u_1, \ldots, u_k, u_{k+1}$ with $k \geq 1$ where (u_0, u_{k+1}) is an edge on the outer-face of the current graph G', and $u_1, \ldots, u_k$ are new vertices that induce a path and have no edges to G' other than (u_0, u_1) and (u_k, u_{k+1}).



(a) An outer-planar graph.



(b) A segment-representation that is not outer-segment at **e**.



(c) A representation as a circle graph that is not order-preserving at **c**.

Figure 6.5: Representations of an outer-planar graph.

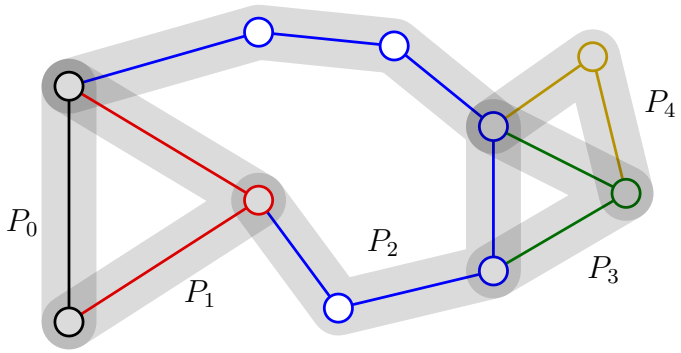


Figure 6.6: Building a 2-connected outerplanar graphs by adding ears.

Thus we iteratively build a representation $\mathcal{R}$ of the subgraph formed by the first few ears. A crucial requirement of $\mathcal{R}$ is the following *order-condition*: If w and w' are the counterclockwise and clockwise neighbours of v on the outer-face, then we encounter the neighbours of v in order, starting with w and ending with w', while walking along $\mathbf{v}$. Put differently, the broken face-vertex-incidence is the one on the outer-face—this is equivalent to linearly order-preserving with the "natural" way of starting and ending in the outer-face. We consider $\mathbf{v}$ to be directed so that it intersects first $\mathbf{w}$ and last $\mathbf{w}'$; the two ends of $\mathbf{v}$ are hence distinguished as head and tail.

The second crucial ingredient for both proofs is to reserve (somewhat similar as was done for faces in Chapter 3 and Chapter 5 and also [31, 42]) for each edge a region that can be used to attach subgraphs. Thus define a *private region* $S_{\mathbf{uv}}$ of an edge (u, v) to be a region that contains an end of $\mathbf{u}$ and an end of $\mathbf{v}$ and does not intersect any other curve or private region of $\mathcal{R}$. Both constructions maintain such a private region $S_{\mathbf{uv}}$ for every outer-face edge (u, v). Moreover, if v is the clockwise neighbour of u, then $S_{\mathbf{uv}}$ contains the tail of $\mathbf{u}$ and the head of $\mathbf{v}$.

6.3.1 Circle-chord representation

We now re-prove that outer-planar graphs are circle graphs, and show that furthermore constructed representation can preserve the order with respect to any given outer-planar embedding.

Theorem 6.5. *Every outer-planar graph G has a cyclically order-preserving representation as an intersection graph of chords of a circle C with respect to any outer-planar embedding of G.*

Proof. It suffices to prove the claim for a 2-connected outer-planar graph G since every outer-planar graph G' is an induced subgraph of a 2-connected outer-planar graph G, and therefore a string representation for G also yields one for G' by deleting curves of vertices in $G - G'$.

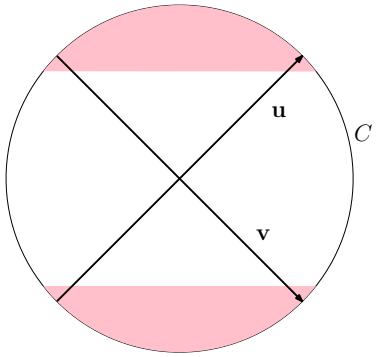
We create a representation $\mathcal{R}$ while building up the graph via adding ears, and maintaining curve directions and private regions as explained before. We maintain the invariant that each private region $S_{\mathbf{uv}}$ is bounded by parts of circle C and a chord of C and does not contain the crossing of $\mathbf{u}$ and $\mathbf{v}$. Further, the tail of $\mathbf{u}$ and the head of $\mathbf{v}$ lie in the interior of the circular arc that bounds $S_{\mathbf{uv}}$.

In the base case, G is an edge (u, v) which can be represented by two chords through the center of C. See Figure 6.7. We reserve two private regions for (u, v), because the outer-face of a single-edge graph should be viewed as containing this edge twice (we can add ears twice at it). All conditions are easily verified.

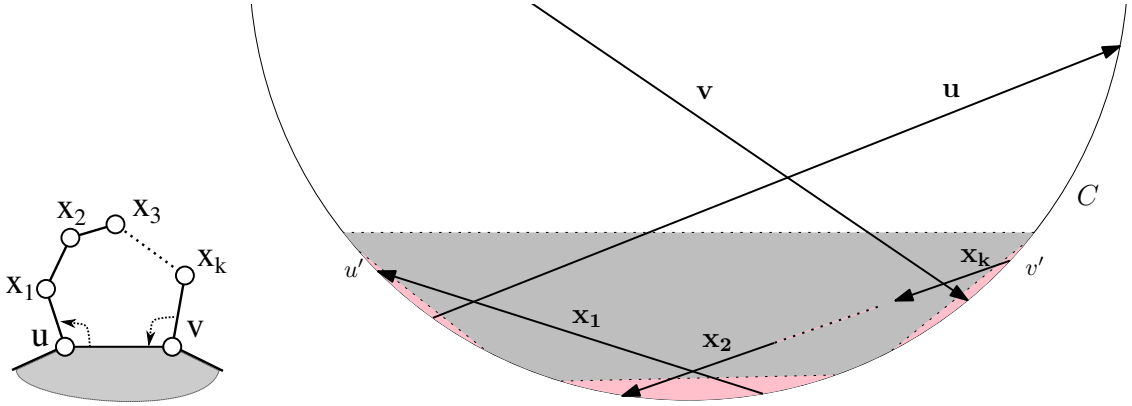
For the induction step, let us assume that G was obtained by adding an ear $P = u, x_1, \ldots, x_k, v$ at some edge (u, v), with u the counter-clockwise neighbour of v on the outer-face. Consider the private region $S_{\mathbf{uv}}$ of edge (u, v) and let $C[u, v]$ be the arc of C between the tail of $\mathbf{u}$ and the head of $\mathbf{v}$ that lies inside $S_{\mathbf{uv}}$. Let u' and v' be two points on C just outside $C[u, v]$ but still within $S_{\mathbf{uv}}$. If $k = 1$, then we add x_1 by using chord $\overline{u'v'}$ for $\mathbf{x_1}$. If $k > 1$, then we insert $2k - 2$ points on the interior of $C[u, v]$ and create chords for $\mathbf{x_1}, \ldots, \mathbf{x_k}$ so that everyone intersects as required. See Figure 6.7, which also shows the private regions that we define for the new outer-face edges.

Since $S_{\mathbf{uv}}$ was convex, all new curves are inside it and do not intersect any other curves. The orientation of these new curves is determined by the order-condition: $\mathbf{x_i}$ should be oriented so that it intersects first $\mathbf{x_{i+1}}$ (where $x_{k+1} := v$) and then $\mathbf{x_{i-1}}$ (where $x_0 := u$). In particular this means that the private region $S_{\mathbf{x_i x_{i+1}}}$ contains the tail of $\mathbf{x_i}$ and the head of $\mathbf{x_{i+1}}$, and hence satisfies the condition on private regions.

It remains to check that the order-condition is satisfied for $\mathbf{u}$. Since $S_{\mathbf{uv}}$ contained the tail of $\mathbf{u}$, this means that $\mathbf{x_1}$ becomes the first curve to be intersected by $\mathbf{u}$, which is correct since x_1 is the clockwise neighbour of u on the outer-face. Likewise one argues that the order-condition holds for $\mathbf{v}$. Hence all conditions hold, and after repeating for all ears we



(a) The base case.



(b) Adding chords for an ear for $k = 2$.

Figure 6.7: Illustration for the proof of Theorem 6.5.

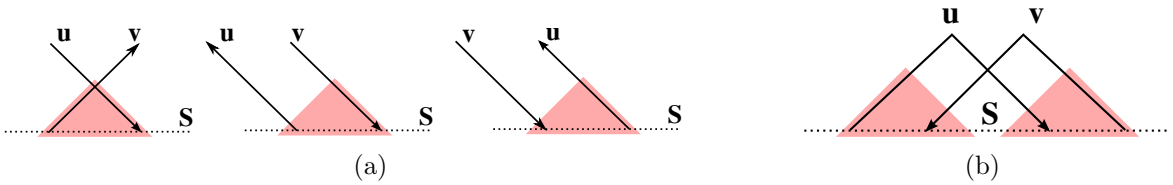


Figure 6.8: Three types of private regions (three more can be obtained by flipping horizontally), and the base case.

obtain an order-preserving representation as an intersection graph of chords of a circle. □

6.3.2 B_1-VPG representation

Now we create, for any outer-planar graph, a B_1-VPG representation that is order-preserving and outer-string. However, the ends will not be on a circle; instead they will lie on a closed curve S that we maintain throughout the construction and that surrounds the entire representation $\mathcal{R}$ without truly intersecting any curve. All vertices are 1-bend poly-lines with slopes ± 1 (after rotating by 45° this gives the B_1-VPG representation); this allows us to use an orthogonal curve for S. Figure 6.8 illustrates types of private regions that we will use for this construction: $S_{\mathbf{uv}}$ contains no bend of **u** or **v**, and it is an isosceles right triangle whose hypotenuse lies on S.

Theorem 6.6. *Every outer-planar graph G has a cyclically order-preserving outer-1-string B_1-VPG-representation $\mathcal{R}$.*

Proof. As before it suffices to prove the claim for 2-connected outer-planar graphs G. We proceed by induction on the number of vertices, building $\mathcal{R}$ while adding ears. In the base case, G is an edge (u, v) which can be represented by two 1-bend curves positioned and oriented as shown in Figure 6.8b, which also shows the private region. We use a horizontal segment for S (this can be expanded into a closed curve surrounding $\mathcal{R}$ arbitrarily).

For the induction step, let us assume that G was obtained by adding an ear $P = u, x_1, \ldots, x_k, v$ at some edge (u, v), with u the counter-clockwise neighbour of v on the

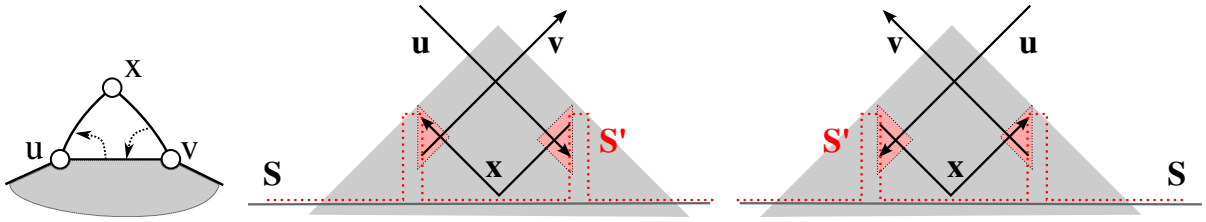


Figure 6.9: Adding a single node if **u** and **v** have different slopes.

outer-face. After possible rotation the hypotenuse of the private region $S_{\mathbf{uv}}$ is horizontal with $S_{\mathbf{uv}}$ above it. We distinguish cases:

1. **u** *and* **v** *have different slopes in* $S_{\mathbf{uv}}$ *and* $k = 1$ *(i.e. we add one vertex* x*).*

   We add a 1-bend curve **x** with the bend pointing downwards. See Figure 6.9, which also shows the private regions that we define for (u, x) and (x, v). Curve **x** fits entirely inside $S_{\mathbf{uv}}$ by placing the bend in the interior of $S_{\mathbf{uv}}$ and shortening **u** and **v** appropriately so that the ends of **x** are vertically aligned with those of **u** and **v**. We can now easily find a new curve S' by adding "detours" to S that reach the hypotenuses of the new private regions. These detours are inside $S_{\mathbf{uv}}$ and hence intersect no other curves (since we shortened **u** and **v**). So the new curve S' is a closed curve that surrounds the new representation as desired.

   The orientation of **x** is again determined by the order-condition, and exactly as in Theorem 6.5 one argues that this respects the order-condition at **u** and **v**, since our choice of curve for **x** ensures that it crosses **u** *after* the crossing of **u** with **v**.

2. **u** *and* **v** *have different slopes in* $S_{\mathbf{uv}}$ *and* $k > 1$ *(i.e. we add at least two vertices* $x_1, \ldots, x_k$*.)*

   We add a path of 1-bend curves $\mathbf{x_1}, \mathbf{x_2}, \ldots, \mathbf{x_k}$ with their bends at the top, and define private regions as illustrated in Figure 6.10. Each curve $\mathbf{x_i}$ is oriented as required by the order-condition, and again one verifies the order-condition for **u** and **v**. We can re-use the same S.

3. **u** *and* **v** *have the same slope inside* $S_{\mathbf{uv}}$*.*

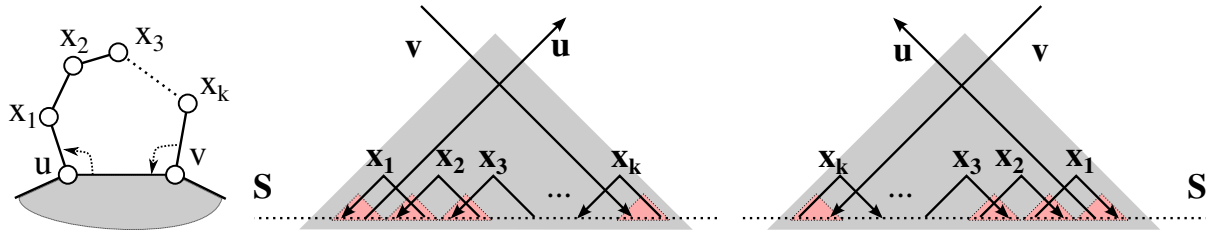


Figure 6.10: Adding 2 or more nodes if **u** and **v** have different slopes.

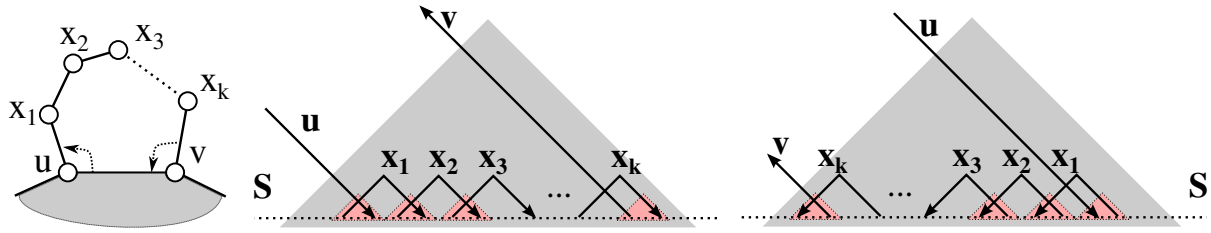


Figure 6.11: Adding one or more vertices if **u** and **v** have the same slope. We only show two of the four possible configurations.

> We add a path of 1-bend curves $\mathbf{x_1}, \mathbf{x_2}, \ldots, \mathbf{x_k}$ (possibly $k = 1$) with their bends at the top, and define private regions as illustrated in Figure 6.11. Each curve $\mathbf{x_i}$ is oriented as required by the order-condition, and one verifies all conditions using the same S.

The final representation of the whole graph is order-preserving due to the order-condition, outer-string due to poly-line S, and B_1-VPG (after a 45°-rotation) since every curve has one bend. □

In our B_1-VPG-representation, every vertex-curve is an ⌞ in one of the four possible rotations ⌞, ⌟, ⌜, ⌝. (All four may be used, since private regions get rotated in Case 1.) It is easy to create representations with ⌞ only if we need not be order-preserving (use ⌒ in Case 1) or need not be outer-string (see also Lemma 6.8), but finding an outer-string order-preserving representation using only ⌞'s remains open.

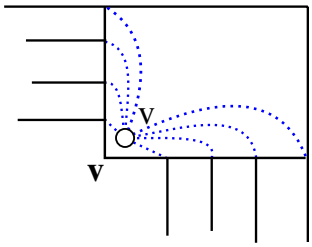


Figure 6.12: Converting an ⌊-contact representation into a planar drawing with the same order.

6.3.3 Beyond outer-planar graphs?

One wonders what other graph classes might have order-preserving 1-string representations, preferably outer-string ones. We study this here for some graph classes. We start with the *series-parallel graphs* that we saw in Section 5.1.2. We need an observation:

Lemma 6.7. *Every graph G with an $\{\llcorner\}$-contact representation has a cyclically order-preserving 1-string representation for some planar embedding of G.*

Proof. For any $\{\llcorner\}$-contact representation with true ⌊'s (neither rotated nor degenerated into a horizontal or vertical line segment), we can create a planar drawing that matches the order of touching-points along each ⌊. Namely, draw a point for v slightly above and to the right of the corner of the bend in $\mathbf{v}$. Connect v to all touching-points on $\mathbf{v}$, and to the two ends of $\mathbf{v}$. Because every curve is an ⌊, the curves whose ends touch $\mathbf{v}$ all come from the left at the vertical segment of $\mathbf{v}$ or from the bottom at the horizontal segment of $\mathbf{v}$. Therefore the added lines do not cross any curves and so give a planar drawing of G that is clearly respected by the representation. Extending the ⌊'s slightly hence gives the desired 1-string representation. □

Since series-parallel graphs have an $\{\llcorner\}$-contact representation (Corollary 5.3), we have:

Corollary 6.8. *Every series-parallel graph G has a 1-string representation using ⌊s only that is cyclically order-preserving for some planar embedding of G.*

It would be interesting to know whether this result can be extended to the planar Laman-graphs, which have a B_1-VPG-contact representation (see Lemma 5.4), but not all $\llcorner$s are necessarily in the same rotation and so it is not clear whether this is cyclically order-preserving. Of particular interest would be planar bipartite graphs, which can even be represented by horizontal and vertical touching line segments (see Lemma 5.1), but again it is not clear how to make this order-preserving in the cyclic model[2].

As for having the representation additionally being outerstring: this is not always possible. Let H be the graph obtained by subdividing every edge in a $K_{2,3}$; one verifies that H is series-parallel. It is easy to see (Lemma 2.6) that H is not outer-string, since $K_{2,3}$ is not outer-planar. So H has no outer-string representation, much less one that is 1-string and order-preserving.

Observation 6.9. *There is a series-parallel graph that is not an outer-string graph.*

Now we turn to partial 3-trees. We showed in Theorem 6.3 that there exist planar 3-trees (hence partial 3-trees) that do not have an order-preserving 1-string representation. We now study some subclasses of partial 3-trees that are superclasses of outer-planar graphs.

Recall the definitions of IO-graphs an Halin graphs from Chapter 5. An *IO-graph* is a planar graph G that has an independent set I such that $G - I$ is a 2-connected outer-planar graph O for which all vertices in I are inside inner faces of O. A *Halin graph* is a graph that consists of a tree T and a cycle C that connects all leaves of T. Both types of graphs are well-known to be partial 3-trees. In Chapter 5, we provided constructions of 1-string $\{\llcorner\}$-representations for both Halin graphs and IO-graphs. Inspection of both constructions shows that these respect the standard planar embedding (where O respectively C is one face). Namely, for IO-graphs, the condition on the IO-private region guarantees that the newly inserted vertex intersects its neighbours in order. For Halin graphs, all outerface vertices have degree 3, and for those any order of neighbours is order-preserving. Inner vertices may have a higher degree, but the order of unfinished rays ensures that they intersect their children in order. We hence have:

[2]The very recent result by Gonçalves et al. [55] also implies that triangle-free planar graphs have a contact representation using only $\llcorner$'s. Hence, all such graphs (and in particular, planar bipartite graphs) do have cyclically order-preserving 1-string representations.

Theorem 6.10. *Every IO-graph and every Halin-graph has a cyclically order-preserving 1-string $\{\llcorner\}$-representation.*

In these constructions, the ends of the strings are not on the outer-face, and we now show that this is unavoidable. This is obvious for Halin-graphs, since the subdivided $K_{2,3}$ is an induced subgraph of a Halin-graph. As for IO-graphs, define the *wheel* W_n to be the graph that consists of a cycle $C = \{v_1, \ldots, v_n\}$ with n vertices and one universal vertex c connected to all of them. Let the *extended wheel-graph* W_n^+ be the wheel-graph W_n with additionally a vertex w_i incident to v_i and v_{i+1} for $i = 1, \ldots, n$ (and $w_{n+1} := w_1$). See also Figure 6.13. Notice that W_n^+ is an IO-graph.

Theorem 6.11. *For $n \geq 7$, the IO-graph W_n^+ has no cyclically order-preserving outer-1-string representation with respect to the embedding shown in Figure 6.13.*

Proof. Assume for contradiction that it did, and consider the induced representation $\mathcal{R}_W$ of W_n. Let the naming of cycle C be such that $\mathbf{c}$ intersects $\mathbf{v_1}, \ldots, \mathbf{v_n}$ in this order. Define as before $\mathbf{u}[v, w]$ (for any 2-path v, u, w) to be the stretch of $\mathbf{u}$ between the intersection with $\mathbf{v}$ and $\mathbf{w}$. Now define R to be the region bounded by $\mathbf{c}[v_1, v_n]$ (which is almost the entire curve $\mathbf{c}$), as well as $\mathbf{v_n}[c, v_1]$ and $\mathbf{v_1}[v_n, c]$ (which exist since (v_1, v_n) is an edge). See also Figure 6.13.

Consider v_i for $i = 3, 4, 5$, which is adjacent to neither v_1 nor v_n. Then $\mathbf{v_i}$ intersects the boundary of R (because it intersects $\mathbf{c}[v_1, v_n]$ by assumption), but does not intersect it twice, else it would intersect $\mathbf{c}$ twice or intersect $\mathbf{v_1}$ or $\mathbf{v_n}$. Hence one end of $\mathbf{v_i}$ is inside R while the other one is outside, and so not both ends of $\mathbf{v_i}$ can be on the outerface for $i = 3, 4, 5$.

This shows that W_n is not outer-1-string in the sense that for some vertex not both ends of the curves are on the outerface. Now consider W_n^+, and the vertices w_3 and w_4 that were added at v_4 when creating W_n^+. Since w_3 and w_4 are adjacent to none of c, v_1, v_n, and since the drawing is outer-string, both $\mathbf{w_3}$ and $\mathbf{w_4}$ (and therefore their intersections with $\mathbf{v_4}$) must be outside R.

So walking along $\mathbf{v_4}$ starting at the end inside R, we encounter $\mathbf{c}$ and then one of $\{\mathbf{w_3}, \mathbf{w_4}\}$. We assume that we encounter $\mathbf{w_3}$ before $\mathbf{w_4}$; the other case is symmetric (and

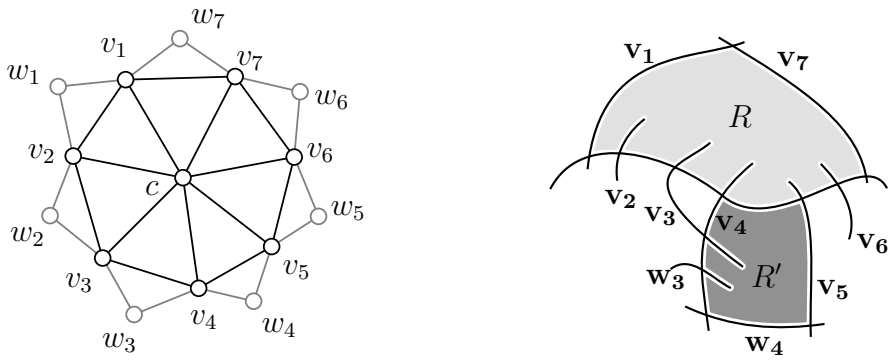


Figure 6.13: An illustration for the proof of Theorem 6.11.

results in $\mathbf{v_5}$ having no end on the outerface). Consider the region R' enclosed by $\mathbf{v_4}[c, w_4]$, $\mathbf{w_4}[v_4, v_5]$, $\mathbf{v_5}[w_4, c]$ and $\mathbf{c}[v_5, v_4]$. Since $\mathbf{w_4}$ is outside R, so is R'. Curve $\mathbf{v_3}$ intersects $\delta R'$, because it intersects $\mathbf{v_4}$, and this intersection must be on $\mathbf{v_4}[c, w_3]$ to preserve the order of edges around v_4 (and since we know that $\mathbf{c}, \mathbf{w_3}, \mathbf{w_4}$ intersect $\mathbf{v_4}$ in this order). Curve $\mathbf{v_3}$ cannot intersect $\delta R'$ again, else it would intersect $\mathbf{c}$ or $\mathbf{v_4}$ twice or would intersect $\mathbf{w_4}$ or $\mathbf{v_5}$, which it shouldn't. Therefore one end of $\mathbf{v_3}$ is inside R', which is outside R. The other end of $\mathbf{v_3}$ is inside R. So neither end of $\mathbf{v_3}$ is on the outerface. Contradiction. □

6.3.4 Order-preserving segment representations

Middendorf and Pfeiffer [74] showed that every B_1-VPG representation that uses only curves of shapes ⌊ and ⌈ can be transformed into a segment representation. We introduced their claim as Lemma 2.5 in Section 2.2.2. Here we prove a stronger claim which also stipulates that the transformation preserves the order of crossings along each curve. This implies that graphs with linearly and cyclically order-preserving ⌊- and ⌈-representations have corresponding order-preserving segment representations.

Lemma 6.12. *Let G be a graph with a $\{⌊, ⌈\}$-representation R. Then there is a string representation S of G such that every curve in S is a line segment. Furthermore, for every vertex $v \in V(G)$, the order of intersections along segment $\mathbf{v}^S$ in S matches the order of*

intersections along curve $\mathbf{v}^R$ *in* R*. The slope of* $\mathbf{v}^S$ *is negative if* $\mathbf{v}^R$ *is an* $\llcorner$ *and positive otherwise.*

Proof. The proof is exactly the one of [74], but we clarify the invariants to argue that order is preserved.

By Lemma 2.4, we can assume that we are given a representation R where all segments have distinct coordinates. We can now order the curves in R according to the x-coordinates of their vertical segments and denote them by $\mathbf{r_1}, \ldots, \mathbf{r_n}$ in this order left to right.

We call a curve in R *right-visible* if its horizontal segment can be extended rightwards "to infinity" without intersecting any other curve. See also Figure 6.14(a). We now prove the lemma by showing the following stronger claim:

Claim. The graph represented by R has a segment representation S enclosed in a rectangular area Θ such that:

- Curve $\mathbf{v}^S$ touches the right border of Θ if and only if $\mathbf{v}^R$ is right-visible. Furthermore, the order of touch points along the right border of Θ matches the order of y-coordinates of the horizontal segments of right-visible curves in R.
- For every vertex $v \in V(G)$, the order of intersections along segment $\mathbf{v}^S$ in S matches the order of intersections along curve $\mathbf{v}^R$ in R.
- No segment of S is vertical. Furthermore, the slope of $\mathbf{v}^S$ is negative if $\mathbf{v}^R$ is an $\llcorner$ and positive otherwise.

We prove this claim by induction on the number of curves in R. For $n = 1$, construct S using an arbitrary non-vertical segment with the appropriate (positive or negative) slope.

For $n > 1$, apply induction to construct a representation S_0 enclosed in a rectangle Θ_0 for the graph represented by $\mathbf{r_1}, \ldots, \mathbf{r_{n-1}}$. We will now construct S by placing segment $\mathbf{r_n}$ into S_0. Observe that all curves that intersect $\mathbf{r_n^R}$ in R are right-visible in $R - \mathbf{r_n^R}$ (since $\mathbf{r_n}$ has maximal x-coordinate and no points to the left of its vertical segment, see also Figure 6.14). Thus, they touch the right border of Θ_0, and the order of the touch points

matches the order of intersections along the vertical segment of $\mathbf{r}_\mathbf{n}^R$. Denote the top-most curve that intersects $\mathbf{r}_\mathbf{n}^R$ by $\mathbf{t}$ and the bottom-most curve by $\mathbf{b}$.

Extend all the segments in S_0 that touch the right border of Θ_0 by a small amount so that they go beyond the border of Θ_0, but do not create any additional intersections among themselves. If $\mathbf{r}_\mathbf{n}^R$ has the shape of an $\llcorner$, insert a segment $\mathbf{r}_\mathbf{n}^S$ with an appropriate negative non-vertical slope so that it intersects all the desired curves (starting with $\mathbf{t}$ and ending with $\mathbf{b}$) to the right of Θ_0. Align the right end of $\mathbf{r}_\mathbf{n}^S$ with the ends of all the other curves and observe that there is new rectangle Θ that encloses S and such that all the right-visible curves of R touch Θ, and all the other conditions hold. Proceed analogously if the shape of $\mathbf{r}_\mathbf{n}^R$ is $\ulcorner$. □

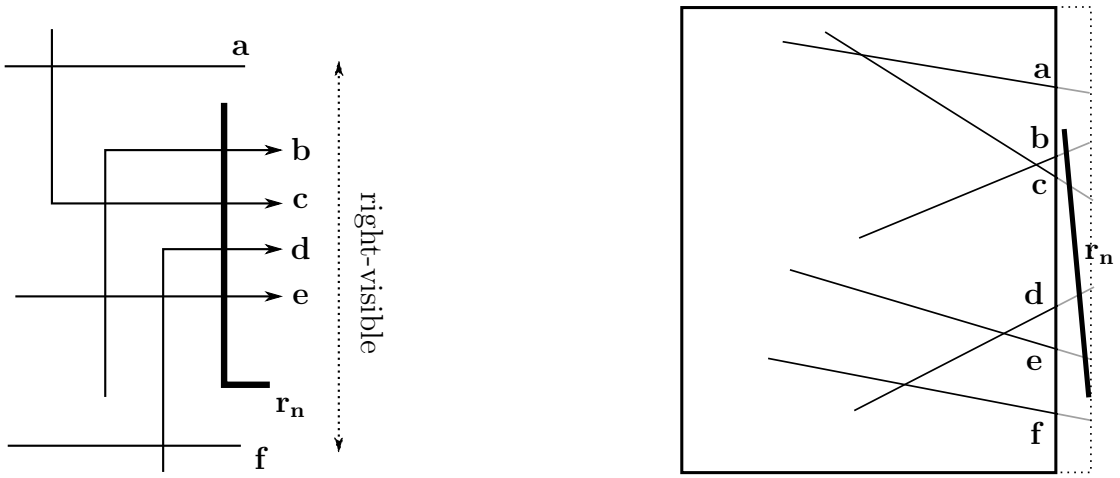
Note that one can adjust to proof to show that if the representation is outer-string, so is the constructed segment representation. Applying Lemma 6.12 to Corollary 6.8 and Theorem 6.10, we get:

Corollary 6.13. *Every series-parallel, IO- and Halin graph has a segment representation that is cyclically order-preserving for some planar embedding.*

The resolution delivered by Lemma 6.12 is very large: with every added vertex, the region to the right of Θ into which we can extend segments without creating intersections gets much narrower, and so we need very steep slopes. In general, this is expected since recognizing SEG is $\exists\mathbb{R}$-hard [69]. But here we are studying a subclass of SEG, namely, graphs that have $\{\llcorner, \ulcorner\}$-representations. Can we find segment representations for them with polynomial-sized coordinates, at least for the special cases listed in Corollary 6.13?

6.4 Selectively order-preserving representations

There are two drawbacks of both the aforementioned order-preserving models. Firstly, neither linear or cyclic order preservation is defined for k-string graphs with $k > 1$. Secondly, string representations derived from contact representations by extending curves are not order-preserving, even though every contact representation gives rise to a very natural embedding of a planar graph. Thus, we propose a third model of selective order-preservation.



(a) (b)

Figure 6.14: (a) An illustration of right-visible curves. (b) Adding $\mathbf{r_n}$ to the segment representation.

Assume that a string representation of a graph is given. Thicken each curve slightly, and consider the cyclic order of intersections while walking around the thickened string. The representation is *selectively order-preserving* with respect to a planar embedding of the graph if the cyclic order of neighbours around a vertex forms a subsequence of the intersections encountered while walking "around" its string. With this, any contact representation becomes a selectively order-preserving 1-string representation after extending the curves a bit. Since this model's restriction is weaker, all our positive results transfer, but the proofs of the negative results no longer hold.

Corollary 6.14. *(1) Planar bipartite graphs have selectively order-preserving segment representations.*

(2) Laman graphs have selectively order-preserving B_1-VPG representations.

(3) Series-parallel, outerplanar, Halin graphs and IO-graphs have selectively order-preserving $\llcorner$-representations and segment representations.

(4) Planar graphs have selectively order-preserving B_3-VPG 2-string representations.

Proof. Planar bipartite graphs have contact representations of segments by Lemma 5.1 [38]. Laman graphs have B_1-VPG-contact representations by [65]. Claim 2 was shown in Lemma 6.12 and Corollary 6.13.

Planar graphs have contact representations using T-shapes [39]. By tracing each T-shape with a curve, we can create a B_3-VPG touching representation and Claim 4 holds. □

We conjecture that not all planar graphs have selectively order-preserving 1-string representations, but this remains open.

6.5 Conclusions

In this chapter, we studied 1-string representations that respect a planar embedding. We have introduced three models of order preservation. The first one takes an arbitrary order of crossing along each curve as an input and asks if a representation with this order of crossings exists. We showed that given such an order, it is possible to decide in linear time if the representation exists. Then we defined an extended model that asks for an order-preserving 1-string representation given a planar embedding of a graph. This is equivalent to asking for a linear order-preserving representation, regardless of where the break point in the rotation scheme around each vertex is. This is the natural model for planar graphs, and the problem is likely NP-hard (see Section 8.5 for the discussion). We showed that such representations exist for outer-planar graphs, series-parallel, IO-graphs and Halin graphs, but there are planar graphs (even planar 3-trees) without such representations. Lastly, we introduced an order-preserving model for representations that are not necessarily 1-string. The model is capable of capturing the order of crossings induced by the contact representations. All planar graph have such representations, but bipartite planar graphs, Laman graphs, series-parallel graphs and Halin graphs have representations using shapes of small complexity.

As for open problems, what other graph classes have cyclically order-preserving 1-string representations? A natural candidate to investigate would be the 2-outer-planar graphs, for which Lemma 6.2 cannot be applied since a triple-stellation is never 2-outer-planar. Other interesting candidates would be planar 4-connected graphs.

Chapter 7

String Representations with Many Crossings

In this chapter, we investigate graphs that have string representations, but in any representation, some curves must intersect more than once. In other words, what graphs are in String, but not in 1-String?

The results in this section are not yet published.

7.1 Exponential construction

While having curves intersect multiple times may be a convenient method of constructing string representations (see for example the construction of string representations for planar graphs by [40], Section 2.2), there are graphs that require such representations, as is implied by the following result of Kratochvíl and Matoušek from 1991 [70].

Theorem 7.1 (Kratochvíl, Matoušek [70])**.** *There are graphs that require an exponential number of crossings in any string representation.*

As we build on top of their construction, we briefly review it here.

Proof of Theorem 7.1. Consider the graph depicted in Figure 7.1 obtained by subdividing a 2×2 grid so that the "middle horizontal path" is formed by vertices A, u_k, u_{k-1}, $v_k, \ldots, u_3$, u_2, v_3, u_1, v_2, u_0, v_1, B. Furthermore, connect every vertex u_i, $i > 0$ to one more subdivision vertex on the lower boundary of the grid, and subdivide the "upward edge" from u_0 with a vertex b. Denote the middle vertex on the top boundary of the grid by a. Let us call this graph H'_k. Notice that since H'_k is a subdivision of a 3-connected planar graph, it has unique embedding in the plane (up to the choice of the outer face). Finally, connect every pair $u_i, v_i, i > 0$ with an edge and call the result H_k. Now define an abstract topological graph (AT-graph—recall that this means that we specify exactly which edges must intersect, see Section 2.2.4) that

- requires intersections of ab with u_iv_i for every $i \leq k$;
- forbids any intersections of u_0b, and any intersection for all edges incident to outer-face vertices with the exception of v_1B and ab
- requires intersections of every edge u_iv_i with some edges in the form of u_rv_s as needed in order to guarantee realizability (see Figure 7.1).[1]

By induction on i, one can now prove that the number of intersections of ab and u_iv_i in such a realization is at least 2^{i-1}.

Recall that At-Realizability reduces to string graph recognition (Lemma 2.7). Applying this, we construct a string graph G (shown in Figure 7.2) for which any string representation can be turned into a correct drawing of the AT-graph H_k by contracting some curves. This contraction adds at most a quadratic number of intersections. Since H_k requires an exponential number of intersection, therefore, so does any string representation of G. □

Recall that in Section 5.3 we investigated graphs with no B_1-VPG representations. Note that specifically for the graph H_k with sufficiently large k shown in Figure 7.1, the realization

[1] We are being purposely vague here. In [70], the authors in fact work with the concept of so-called *weak* At-Realizability where the intersections are allowed, but not required. Even when "allowing but not requiring" all the possible intersections between edges u_iv_i and u_rv_s, $i, r, s \leq k$, 2^{i-1} intersections between ab and u_iv_i are still necessary.

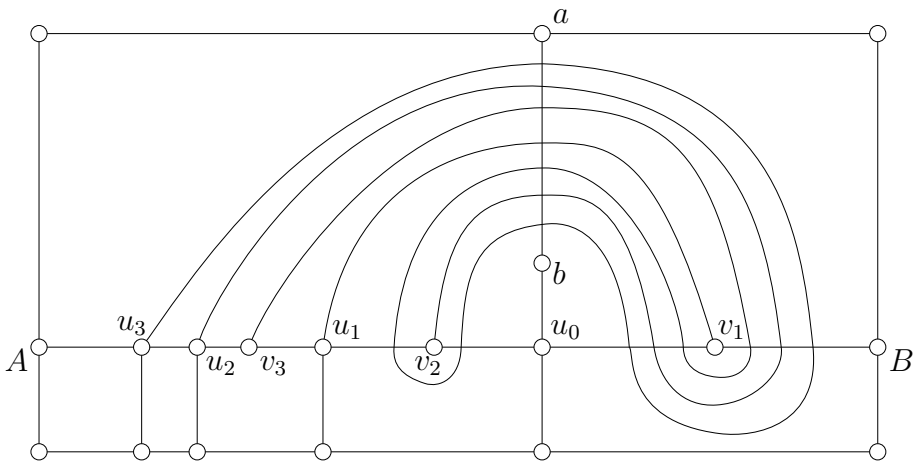
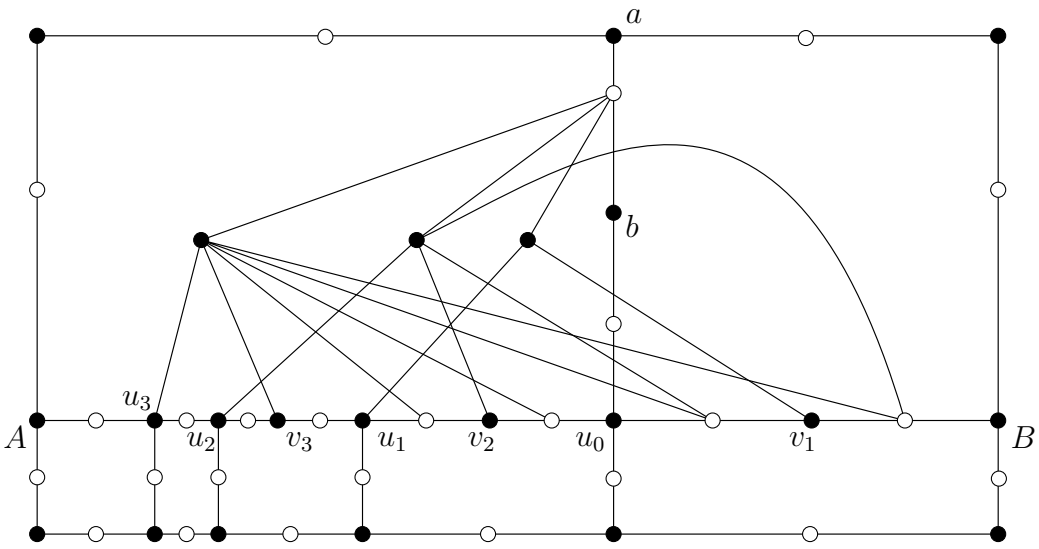


Figure 7.1: An AT-graph H_k that requires an exponential number of intersections in its realization. We show the graph for $k = 3$.

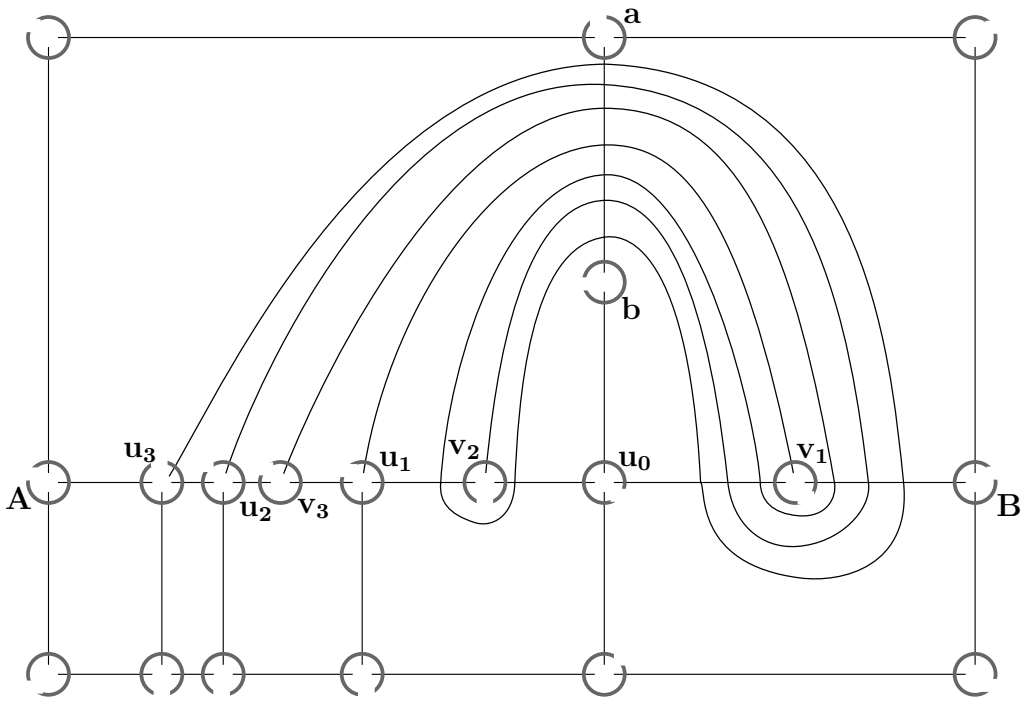
requires 4 crossings between edges (u_3, v_3) and (a, b), hence the graph is not 3-string. As two B_1-VPG curves cannot intersect more than twice, we have B_1-VPG $\subseteq$ 2-String and so the graph is not a B_1-VPG graph. Unfortunately, and as expected, this graph is not planar ($\{u_1, v_1, u_2, v_2, u_3, v_3\}$ can be used to find a $K_{3,3}$-minor; we proved in Theorem 3.1 that all planar graphs are 1-string graphs), but we will use a similar graph later to create a 1-planar graph without a B_1-VPG representation.

7.2 Outer-string graphs

The representation in Figure 7.2 is not outer-string and one can argue that the graph has no outer-string representation. This poses the question of whether there are outer-string graphs that require an exponential number of crossings in any outer-string representation. As there are algorithms that utilize outer-string representations (see [63, 61] and also Chapter 4), a negative result would be extremely interesting from an algorithmic perspective. In this section, we answer the question affirmatively. Thus, we present a construction for outer-string representations inspired by the one of Kratochvíl and Matoušek which shows that exponential size is sometimes required.



(a) The graph G_3 derived from H_3 in Figure 7.1 by applying the reduction from AT-Realizability to recognition of string graphs (Lemma 2.7).



(b) A string representation of G_3 derived from the AT-realization of H_3 in Figure 7.1.

Figure 7.2: A graph with no 3-string representation.

First, we show that there is a correspondence between outer-string representations of a graph, and string representations of what we call its "subdivided apex graph." During our construction, we will be working with subdivided apex graphs, which will make our arguments simpler. Let G be a graph. The *apex graph* H of G is the graph obtained from G by adding a new vertex a connected to all vertices in G. The *subdivided apex graph* of G, denoted by G^+, is obtained from H by subdividing every edge incident to a. See Figure 7.3.

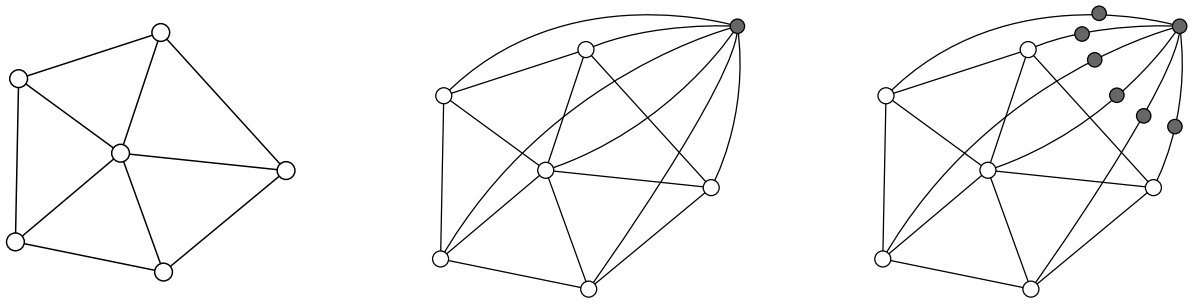
The following characterization is very simple, but surprisingly enough appears to be unknown. The closest related result is by Kratochvíl [67] who argued that a graph G is outer-string if and only if all supergraphs H where $H - G$ is a clique are string graphs.

Lemma 7.2. *Graph G is an outer-string graph if and only if its subdivided apex graph G^+ is a string graph.*

Proof. Assume that we have an outer-string representation R of G. Then we can add a curve $\mathbf{a}$ for the apex vertex into the exterior of R and connect it to the endpoint of every curve of R with a curve that also lies in the exterior of R. This is a string representation of G^+.

For the converse, let R' be a string representation of G^+. All the neighbours of the apex vertex a have degree 2 (those are the subdivision vertices), and we can therefore assume that their curves intersect $\mathbf{a}$ exactly once [68, p. 68]. So, curve $\mathbf{a}$ has precisely one intersection with the curves of its neighbours.

Consider some vertex w of G and the place where $\mathbf{w}$ intersects $\mathbf{s_w}$, where s_w is the subdivision vertex of edge (w, a) of the apex-graph (note that the intersection of s_w and w can be assumed to be unique as s_w has degree 2). At this point, bend and re-reroute $\mathbf{w}$ along $\mathbf{s_w}$ to create a contact with $\mathbf{a}$. Formally, consider the boundary B of the set of points S with distance at most ε to $\mathbf{s_w}$ for a sufficiently small ε. Bend $\mathbf{w}$ at its first intersection with B and continue on B to the closest point $B \cap \mathbf{a}$. Then continue on $\mathbf{a}$ to the other intersection $B \cap \mathbf{a}$, and bend and lead $\mathbf{w}$ on B back to the other intersection of $B \cap \mathbf{w}$. This creates no new intersections since $\mathbf{s_w}$ is only adjacent to $\mathbf{a}$ and $\mathbf{w}$. Since every curve now attaches to $\mathbf{a}$, thickening $\mathbf{a}$ and taking the boundary of the object creates a closed disk D with the entire string representation of G outside of D and every curve intersecting



(a) (b) (c)

Figure 7.3: (a) A graph G. (b) The apex graph of G. (c) The subdivided apex graph G^+.

D. Flipping D inside out, i.e., treating it as an outerface of the resulting combinatorial structure, and deleting $\mathbf{a}$ produces a representation of G where every string $\mathbf{w}$ has some point on the outerface. Tracing around $\mathbf{w}$ (in the sense similar to the above), we can replace it with another curve that intersects the same set and has an end on the outerface, so this gives an outer-string representation of G. □

We first wish to point out a corollary of the results presented in this section. While it was long known that string graph recognition is NP-hard [68], proving that it is in NP was a long-standing open question until proved by Schaefer [78]. Since, for any graph G, we can construct the subdivided apex graph G^+ in polynomial time and the test whether G^+ is a string graph is in NP, the problem of outer-string graph recognition lies in NP.

Corollary 7.3. *The recognition problem of outer-string graphs is in NP.*

Now we construct a graph that requires many intersections in any string representation. This is not a new result (see Theorem 7.1), but our graph is different, and can be used to prove a similar result for outer-string representations later. Fix an arbitrary integer k, and set $K = 20k + 30$. Let $C = c_0, c_1, \ldots, c_{K-1}$ be a cycle of length $20k + 30$ (refer to Figure 7.6a).[2] We assign labels $u_0, \ldots, u_k$, $v_0, \ldots, v_k$ to the vertices of C as follows. Set $u_0 := c_0$, $u_1 := c_{10}, v_0 := c_{20}, v_1 := c_{40}$. For any $i > 1$, let ℓ be such that $v_{i-1} =$

[2]We use this cycle length for ease of notation; a cycle of length $8k + 8$ would be sufficient.

c_ℓ. Then, set $u_i := c_{\ell-10}$ and $v_i := c_{\ell+20}$. In other words, the order of vertices along cycle is $u_0, u_1, v_0, u_2, v_1, u_3, v_2, \ldots, u_i, v_{i-1}, u_{i+1}, v_i, \ldots, u_k, v_{k-1}, v_k$, with 9 other vertices of C between any two of them.

For every pair of vertices (u_i, v_i), we add two new vertices x_i, y_i so that

- $x_i y_i$ is an edge;
- x_i is connected to u_i and y_i is connected to v_i; and
- y_j is connected to every x_i for $i < j$.

Let us call the resulting graph G_k. The subdivided apex graph of G_k is denoted by G_k^+, the apex vertex by a, the common neighbours of a and v_i (or u_i) by s_i^v (or s_i^u, respectively). Figure 7.6b illustrates an outer-string representation of G_k, which can be converted into a string representation of G_k^+ (see Lemma 7.2). Note that $\mathbf{y_k}$ and $\mathbf{x_0}$ intersect 2^{k-1} times. We now argue that this is required.

Theorem 7.4. *In any string representation of G_k^+, curve $\mathbf{y_k}$ intersects curve $\mathbf{x_0}$ at least 2^{k-1} times.*

Proof. Fix a string representation of G_k^+. Delete from it all strings of subdivision vertices between the apex vertex a and c_{2i+1}, for some i; these will not be needed. In consequence, $\mathbf{c_{2i+1}}$ now intersects only two other strings ($\mathbf{c_{2i}}$ and $\mathbf{c_{2i+2}}$) and as before, we may hence assume that $\mathbf{c_{2i+1}}$ has exactly two such intersection points and no more [68, p. 68]. So, for any j, we have a unique point in $\mathbf{c_j} \cap \mathbf{c_{j+1}}$ (addition for all vertices in C is mod K).

Recall that $\mathbf{u}[v, w]$ denotes the stretch of $\mathbf{u}$ between the intersection with $\mathbf{v}$ and $\mathbf{w}$ and define the closed curve $\mathbf{C}$ to be $\bigcup_{j=1}^{K} \mathbf{c_j}[c_{j-1}, c_{j+1}]$. Observe the following:

- The curve $\mathbf{a}$ of the apex vertex is disjoint from $\mathbf{C}$ and hence resides inside or outside. By symmetry, we may assume that $\mathbf{a}$ is outside $\mathbf{C}$.
- For any $i \leq j$, there must exist at least one point in $\mathbf{x_i} \cap \mathbf{y_j}$ since (x_i, y_j) is an edge. We claim that any such point is inside $\mathbf{C}$. If it were outside $\mathbf{C}$, then we could find an outer-planar drawing of K_4 as follows (see Figure 7.4):

- Follow curve $\mathbf{x_i}$ from $\mathbf{x_i} \cap \mathbf{y_j}$ until the nearest point in $\mathbf{x_i} \cap \mathbf{u_i}$. This point may or may not be on $\mathbf{C}$; if it is not then follow $\mathbf{u_i}$ from here to the nearest point on $\mathbf{C}$. Place vertex A here and note that $A \in \mathbf{u_i} \cap \mathbf{C}$. Also, A is connected to $\mathbf{x_i} \cap \mathbf{y_j}$ along a curve within $\mathbf{x_i} \cap \mathbf{u_i}$ that does not cross $\mathbf{C}$.
- Similarly, follow $\mathbf{y_j}$ to $\mathbf{y_j} \cap \mathbf{v_j}$ and (if needed) along $\mathbf{v_j}$ until a point on $\mathbf{C}$. Place B here and note that $B \in \mathbf{v_j} \cap \mathbf{C}$ and B connects to $\mathbf{x_i} \cap \mathbf{y_j}$ along a curve within $\mathbf{y_j} \cap \mathbf{v_j}$ that does not cross $\mathbf{C}$.
- Say $u_i = c_{2s}$ for some s and $v_j = c_{2t}$ for some t. We have $|2t - 2s| \geq 10$. Let s_{2s+2} and s_{2t+2} be the subdivision vertices of edges (a, c_{2s+2}) and (a, c_{2t+2}). Follow $\mathbf{a}$ from $\mathbf{a} \cap \mathbf{s_{2s+2}}$ to an $\mathbf{a} \cap \mathbf{s_{2t+2}}$ and extend the curve along $\mathbf{c_{2s+2}}$ to a point D on $\mathbf{C} \cap \mathbf{c_{2s+2}}$. Also extend the curve along $\mathbf{c_{2t+2}}$ to a point E on $\mathbf{C} \cap \mathbf{c_{2t+2}}$.
- We now have vertices A, D, B, E on curve $\mathbf{C}$, and they occur in this order since A, D, B, E lie on $\mathbf{c_{2s}}, \mathbf{c_{2s+2}}, \mathbf{c_{2t}}, \mathbf{c_{2t+2}}$, and $\mathbf{C}$ visits these strings in order of index. So, we can use $\mathbf{C}$ to draw a circle $A - D - B - E - A$. But we also have a curve from A to B (along $\mathbf{x_i}$ and $\mathbf{y_j}$) and a curve from D to E (along $\mathbf{a}$). Both curves are outside $\mathbf{C}$ if $\mathbf{a}$ and $\mathbf{x_i} \cap \mathbf{y_j}$ are outside $\mathbf{C}$, leading to an outer-planar drawing of K_4, a contradiction.

- Thus for any $\mathbf{x_i}$, at least part of its curve is inside $\mathbf{C}$. But it also must intersect $\mathbf{a}$, so it must have points outside $\mathbf{C}$. So, $\mathbf{x_i}$ must intersect $\mathbf{C}$, which is possible only at $\mathbf{u_i}$. Similarly $\mathbf{y_j}$ intersects $\mathbf{v_j}$ at a point on $\mathbf{C}$ for all j.

- As we walk along $\mathbf{C}$, the intersections with curves in $\{\mathbf{x_i}, \mathbf{y_j}\}$ occur in the same order as cycle C contains the corresponding neighbours, i.e., the order is $\mathbf{x_0}$, $\mathbf{x_1}$, $\mathbf{y_0}$, $\mathbf{x_2}$, $\mathbf{y_1}$, $\mathbf{x_3}$, $\mathbf{y_2}, \ldots, \mathbf{x_k}$, $\mathbf{y_{k-1}}$, $\mathbf{y_k}$. This holds because for each $v \in \{x_i, y_j | 0 \leq i, j \leq k\}$ there is exactly one $w \in \mathbf{C}$ adjacent to v, so $\mathbf{v}$ must cross $\mathbf{C}$ exactly at $\mathbf{w}$, and $\mathbf{C}$ lists the curves of C in order.

Since all relevant intersections happen inside $\mathbf{C}$, we will in the following ignore all parts of curves outside $\mathbf{C}$. Now we are almost ready to prove by induction on i that $\mathbf{y_i}$ intersects $\mathbf{x_0}$ at least 2^{i-1} times, but we need to show a slightly stronger claim for the induction to work. Let R'_i be any representation that satisfies the following:

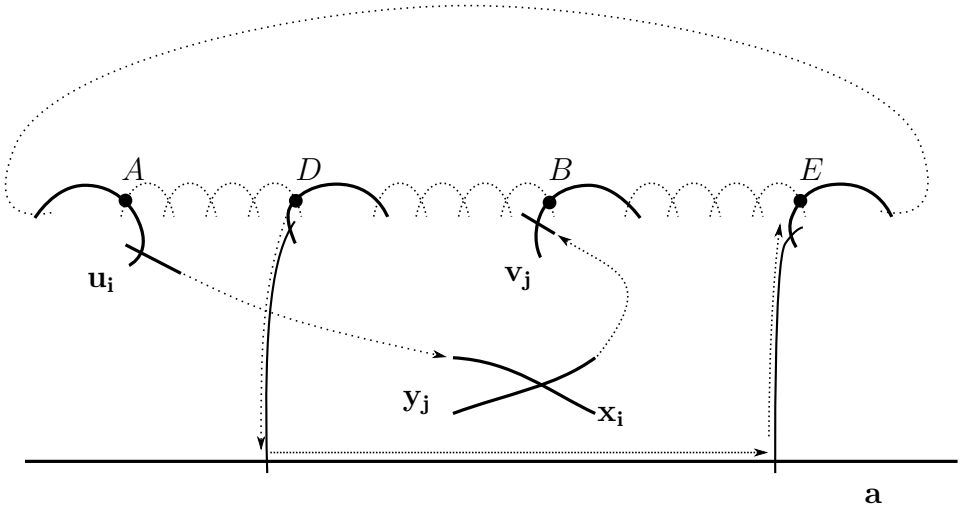


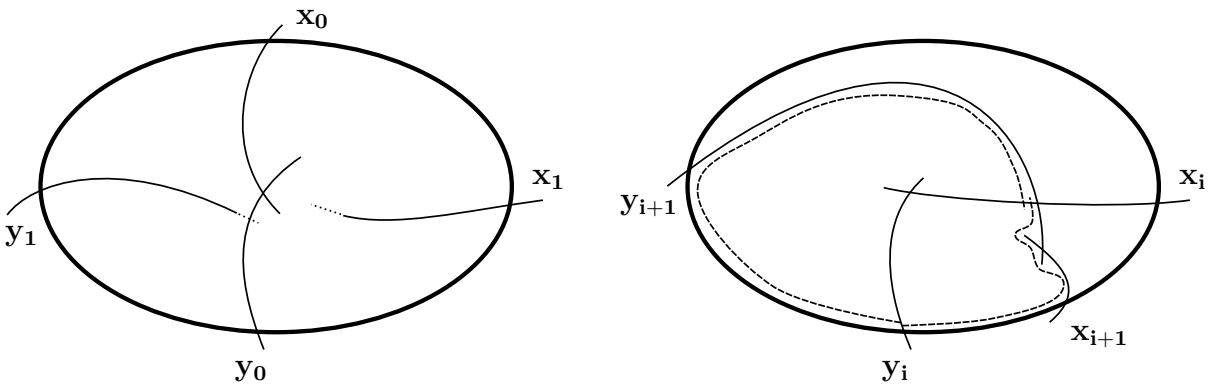
Figure 7.4: An illustration of finding the outer-planar embedding of K_4 for the proof of Theorem 7.4.

- It has a cycle $\mathbf{C}$ with all curves on or inside it.
- It has curves $\mathbf{x_j}$ and $\mathbf{y_j}$ (for $0 \leq j \leq i$) that intersect $\mathbf{C}$ in order $\mathbf{x_0}$, $\mathbf{x_1}$, $\mathbf{y_0}$, $\mathbf{x_2}$, $\mathbf{y_1}$, $\mathbf{x_3}$, $\mathbf{y_2}, \ldots, \mathbf{x_i}$, $\mathbf{y_{i-1}}$, $\mathbf{y_i}$.
- Curves $\mathbf{x_j}$ and $\mathbf{y_j}$ intersect for all $0 \leq j \leq i$.
- There may or may not be intersections of $\mathbf{y_j}$ with $\mathbf{x_r}$ for $r < j$.
- No other curves intersect.

Note that R induces such a representation by omitting strings $\mathbf{x_{i+1}}, \mathbf{y_{i+1}}, \ldots, \mathbf{x_k}, \mathbf{y_k}$ and everything outside $\mathbf{C}$. We now show the following claim:

Claim 7.5. *In any such representation R'_i, curve $\mathbf{y_i}$ intersects $\mathbf{x_0}$ at least 2^{i-1} times.*

First, consider the base case $i = 1$ (see Figure 7.5(a)). The order in which curves intersect $\mathbf{C}$ is $\mathbf{x_0}, \mathbf{x_1}, \mathbf{y_0}, \mathbf{y_1}$, and their combined curve $\mathbf{x_0} \cup \mathbf{y_0}$ splits $\mathbf{C}$ into two parts. Curves $\mathbf{x_1}$ and $\mathbf{y_1}$ intersect $\mathbf{C}$ in different parts. To create an intersection point $\mathbf{x_1} \cap \mathbf{y_1}$,



(a) The base case.

(b) Two possible routes for $\mathbf{y_i}$ the induction step.

Figure 7.5: In the base case, $\mathbf{y_1}$ must cross $\mathbf{x_0}$. In the induction step, a route for $\mathbf{y_{i+1}}$ gives two possible routes for $\mathbf{y_i}$ to $\mathbf{x_i}$.

one of them must cross paths $\mathbf{y_0} \cup \mathbf{x_0}$. Such a crossing must be between $\mathbf{y_1}$ and $\mathbf{x_0}$ (no other crossings are allowed). So, $\mathbf{y_1}$ intersects $\mathbf{x_0}$ at least once as desired.

Assume now that the claim holds for some i. Curve $\mathbf{y_{i+1}}$ is separated from curve $\mathbf{x_{i+1}}$ by $\mathbf{x_i} \cup \mathbf{y_i}$. Thus, curve $\mathbf{y_{i+1}}$ has to intersect $\mathbf{x_i}$ on its way to $\mathbf{x_{i+1}}$. On the way to $\mathbf{x_i}$, it has to create at least 2^{i-1} intersections with $\mathbf{x_0}$, otherwise we could re-route $\mathbf{y_i}$ and use fewer crossings between $\mathbf{y_i}$ and $\mathbf{x_0}$. More precisely (refer to Figure 7.5(b)), $\mathbf{y_i}$ could be re-routed to stay in the proximity of the cycle $\mathbf{C}$ until it reaches $\mathbf{v_{i+1}}$ (follow curves $\mathbf{v_i} = \mathbf{c_j}, \mathbf{c_{j+1}} \dots \mathbf{c_s} = \mathbf{v_{i+1}}$), and then follow $\mathbf{y_{i+1}}$ until reaching $\mathbf{x_i}$. Along this new route (following $\mathbf{y_{i+1}}$) curve $\mathbf{y_i}$ might intersect neighbours of y_{i+1}, but all those neighbours are allowed to be neighbours of y_i as well, so this is a valid representation with less than 2^{i-1} points in $\mathbf{y_i} \cap \mathbf{x_0}$. This contradicts the induction hypothesis.[3] So, $\mathbf{y_{i+1}}$ intersects $\mathbf{x_0}$ at least 2^{i-1} times on the way from $\mathbf{C} \cap \mathbf{y_{i+1}}$ to $\mathbf{y_{i+1}} \cap \mathbf{x_i}$.

On the way from $\mathbf{x_i}$ to $\mathbf{x_{i+1}}$, curve $\mathbf{y_{i+1}}$ needs to create another 2^{i-1} crossings with $\mathbf{x_0}$, otherwise we could re-route $\mathbf{y_i}$ and use fewer crossings as follows: $\mathbf{y_i}$ stays in the proximity of the cycle curves until it reaches $\mathbf{u_{i+1}}$ (follow curves $\mathbf{v_i} = \mathbf{c_j}, \mathbf{c_{j-1}} \dots \mathbf{c_s} = \mathbf{u_{i+1}}$), and then follows $\mathbf{x_{i+1}}$ and $\mathbf{y_{i+1}}$. Thus $\mathbf{y_{i+1}}$ crosses $\mathbf{x_0}$ at least 2^i times as desired. □

[3] It is true, but not obvious, that along the new route, $\mathbf{y_i}$ *must* cross all of $\mathbf{x_{i-1}}, \dots, \mathbf{x_1}$ as well. Rather than arguing this, we switched to the representation R'_i where such crossings are allowed, but not required.

In consequence, we have:

Theorem 7.6. *For any $k \geq 1$, there exists a graph G_k with $O(k)$ vertices that has an outer-string representation, but any outer-string representation of G_k requires two strings to intersect at least 2^{k-1} times.*

Proof. We use graph G_k with $22k + 32$ vertices as defined earlier. By Theorem 7.4, any string representation of G_k^+ requires at least 2^{k-1} intersections between y_k and x_0. Since any such representation can be obtained from an outer-string representation of G_k without changing any string of G_k (see the proof of Lemma 7.2), any outer-string representation of G_k requires at least 2^{k-1} intersections between $\mathbf{y_k}$ and $\mathbf{x_0}$. □

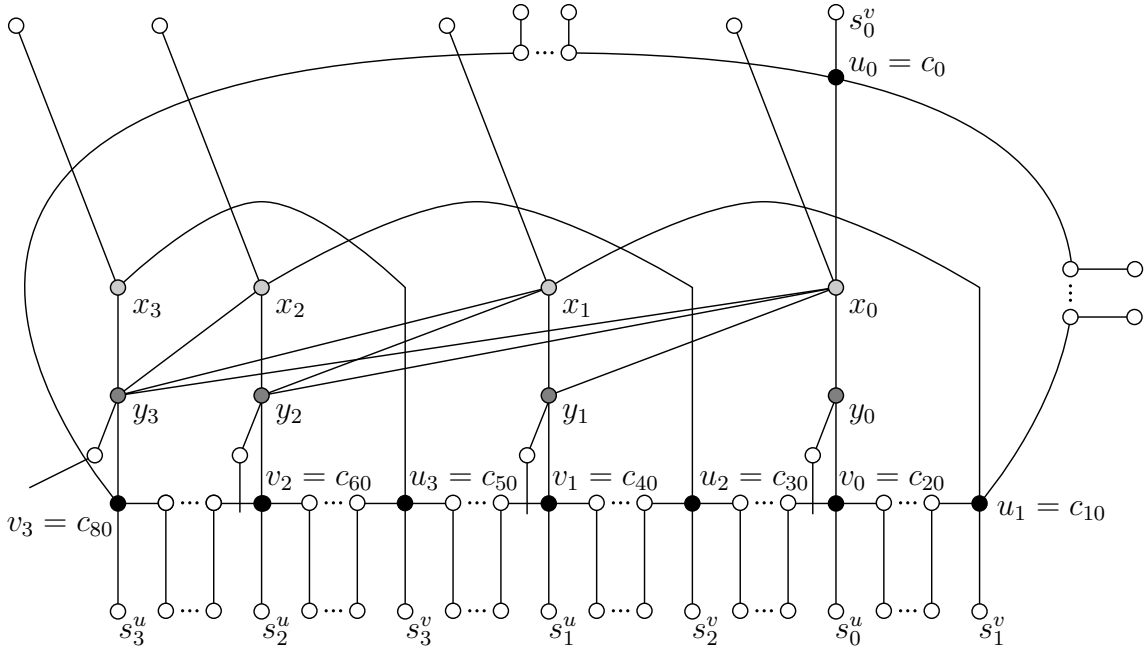
7.3 1-planar graphs

In the previous section, we presented graphs where any outer-string representation must have curves that intersect more than once. Here we show that some of those graphs (and even their subdivided apex graphs) are 1-*planar graphs*, i.e., they can be drawn in the plane such that every edge is crossed at most once. Therefore, 1-planar graphs are not in 1-String. This is an interesting result because the class of 1-planar graphs is very close to the class of planar graphs, which is a subclass of 1-String.

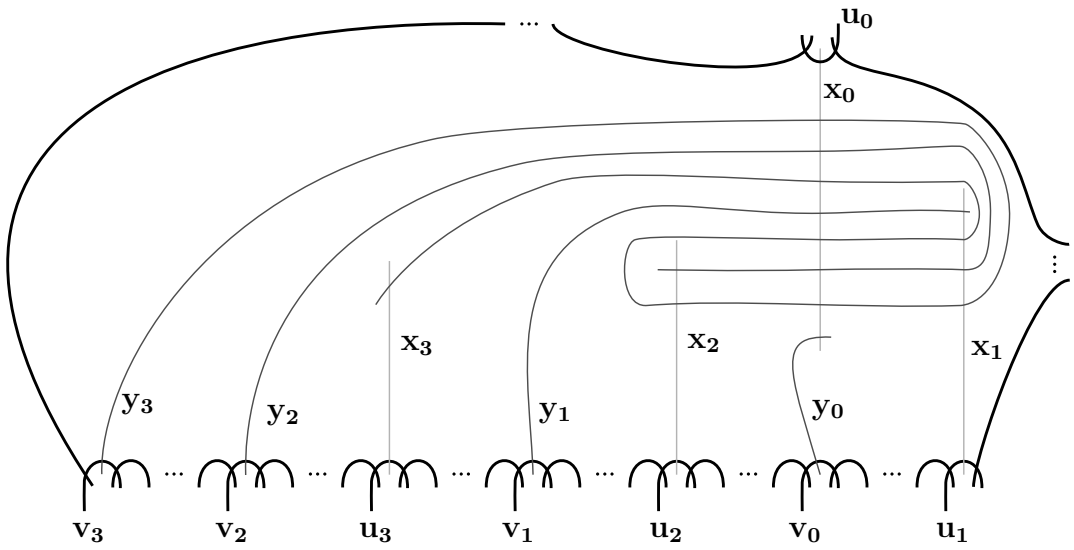
Theorem 7.7. *There are 1-planar graphs that are string graphs but do not have 1-string representations.*

Proof. Let G_k^+ be defined as in the previous section and recall that they are string graphs. By Theorem 7.4, any string representation of graph G_2^+ requires that $\mathbf{y_2}$ and $\mathbf{x_0}$ intersect at least twice. A 1-planar embedding of G_2^+ is shown in Figure 7.7. □

Theorem 7.7 raises many questions about string representations of 1-planar graphs, which is a graph class that has been paid a lot of attention recently. For instance, Thomassen in 1988 characterized 1-planar graphs that have straight line 1-planar drawings [81]. Di Giacomo, Liotta and Montecchiani [52] researched the straight line drawings of 1-planar



(a) The graph G_3^+. The apex vertex a is not shown.



(b) The outer-string representation of G_3.

Figure 7.6: A representation of an outer-string graph that requires exponentially many crossings and the corresponding subdivided apex graph.

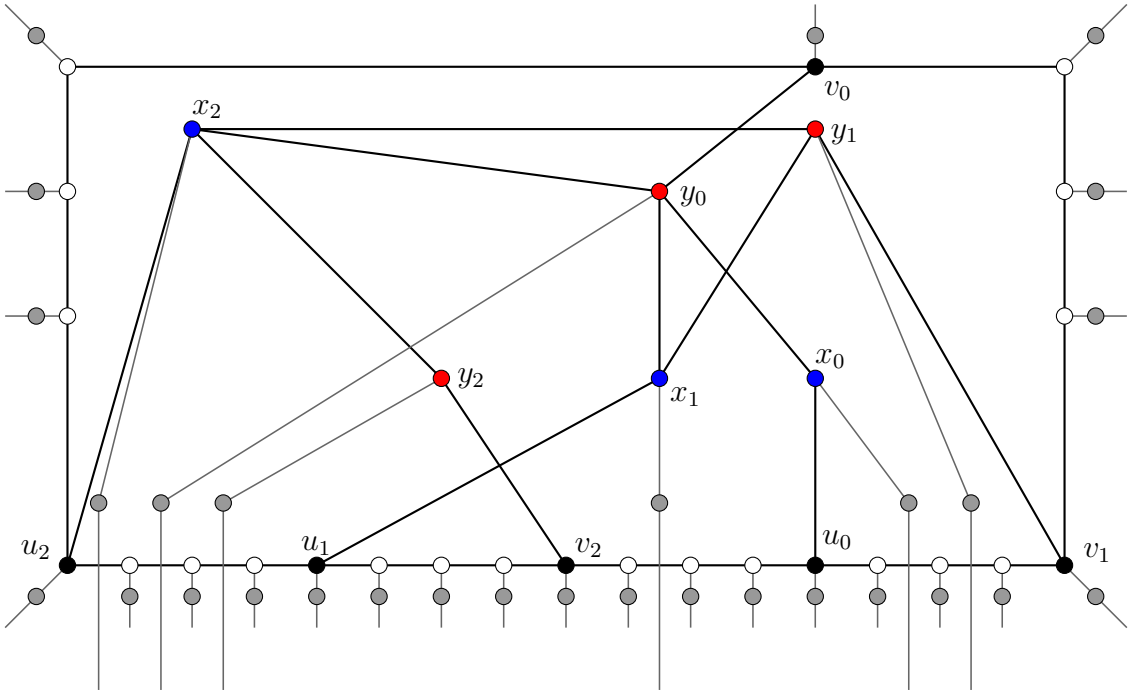


Figure 7.7: A 1-planar embedding of G_2^+. The apex vertex is not shown, but all its incident edges end on the outer-face, so it can be added without more crossings.

graphs with limited number of edge slopes. The visibility representations of 1-planar graphs were investigated e.g. by Brandenburg [22] and Biedl, Liotta, and Montecchiani [16]. For work on the recognition algorithms, see e.g. Auer et al. [9]. A recent survey on 1-planar graphs was published by Kobourov et al. [64]. We are not aware of any research on string representations of 1-planar graphs so far, and none of the aforementioned results seems to trivially imply the existence of string representations for 1-planar graphs, or the opposite. In the rest of this section, we provide some initial results on this topic.

First, we should point out that while any planar graph is a string graph, this is not the case for 1-planar graphs: a full subdivision of any graph, such as K_5, that is not planar but can be drawn with 1 crossing is a 1-planar graph, but not a string graph (see Lemma 2.1). Furthermore, by subdividing the edges of any non-planar graph sufficiently many times, one can obtain a 1-planar graph. As the property of "not having a string representation" is closed under taking subdivisions (recall Observation 2.3), any 1-planar graph obtained by subdividing edges of a graph that is not a string graph is not a string graph either.

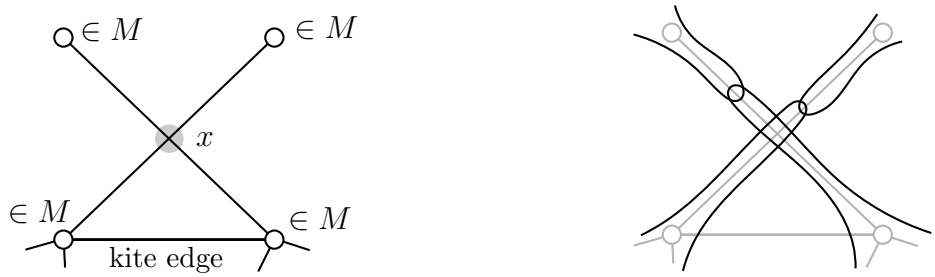
Observation 7.8. *There are 1-planar graphs without string representations.*

Since we know that some, but not all 1-planar graphs have string representations, one naturally wonders what graphs are simultaneously string and 1-planar, but not planar.

Let H be a 1-planar graph with a fixed 1-planar embedding (i.e., the rotation scheme and edge crossings), and let x be a crossing in H. Let M be the four endpoints of the edges involved in the crossing x. The subgraph $H[M]$ induced by M in H is called a *kite* of x if it is isomorphic to K_4. A non-crossed edge of $H[M]$ is called a *kite edge* of the crossing x. See also Figure 7.8.

Theorem 7.9. *Let G be a 1-planar graph. If G has a 1-planar embedding such that every crossing has a kite edge, then G is a string graph.*

Proof. First, note that if G has an embedding with a kite edge for every crossing, it has another such embedding in which no kite edge is crossed. Such an embedding can be constructed be rerouting the crossed kite edges through the vicinity of their crossings. Observe that such a rerouting does not remove any of the kite edges and does not create any new crossing, so the resulting embedding Γ still contains a kite edge for every crossing.



(a) A crossing with a kite edges in a 1-planar graph.

(b) Using the kite edge to construct a string representation.

Figure 7.8: Constructing a string representation of a 1-planar graph with kite edges by tracing along the edges in the vicinity of a crossing. As the kite is represented at the crossing, it does not need to be traced again.

Having constructed Γ, we can construct the string representation of G by tracing along edges in incident to each vertex in Γ (similarly to the construction of Figure 2.1 for planar graphs). In order to trace around edges that cross, the presence of a kite edge uv allows the curves $\mathbf{u}, \mathbf{v}$ to intersect (see Figure 7.8). Curves $\mathbf{u}$ and $\mathbf{v}$ can thus reach the neighbours that they are connected to via a crossed edge. □

Note that the representation constructed in the proof of Theorem 7.9 is a 4-string representation with no restrictions on the shape of the curves. We showed in Theorem 7.7 that there are 1-planar string graphs that do not have 1-string representations. One could ask if the presence of a kite edge can guarantee that a 1-planar graph has a 1-string representation. This is not the case. Figure 7.9(a) shows a 1-planar graph with a single kite edge for every crossing. Note that the graph is very similar to the graph in Figure 7.2a (for $k = 2$), but some edges are missing.

Lemma 7.10. *The graph G depicted in Figure 7.9 is a string graph with a 1-planar embedding where all crossings have a kite edge, but G does not have a 1-string representation.*

Proof. A 1-planar embedding of the graph with kite edges is shown in Figure 7.9(a). Now fix any string representation of $G - x$. Observe that $G - x$ is a full subdivision of a 3-connected

planar graph. By contracting the curves that represent vertices of degree 3 in $G - x$ into points, we obtain a planar drawing of a subdivision of a 3-connected graph which is unique by Whitney's theorem [86]. Thus, the "faces" of the string representation in Figure 7.9(b) can be found in any string representation of $G - x$. Especially, in any representation, curves $\mathbf{u_1}, \mathbf{v_1}$ and $\mathbf{w_1}$ are separated from each other by paths formed by segments of curves $\mathbf{a}, \mathbf{b}, \mathbf{c}, \mathbf{y}, \mathbf{h}, \mathbf{i}, \mathbf{j}$ and $\mathbf{a}, \mathbf{b}, \mathbf{c}, \mathbf{d}, \mathbf{e}, \mathbf{f}, \mathbf{g}$. Curve $\mathbf{x}$ has to cross each of the paths at least once and the only allowed crossing point is the segment of $\mathbf{c}$. Thus, $\mathbf{c}$ and $\mathbf{x}$ intersect at least twice. □

Thus, the presence of kite edge is not sufficient to guarantee the existence of 1-string representations. However, we can show that if all kite edges are present, then we can provide a constant upper bound on the number of bends and intersections needed.

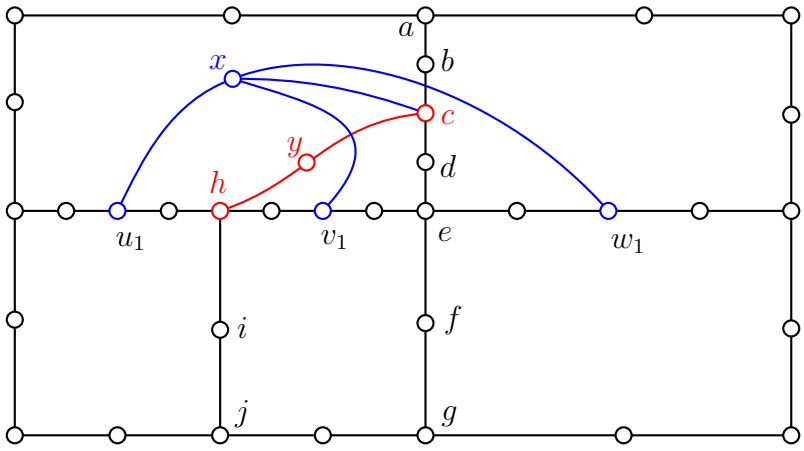
Theorem 7.11. *Let G be a 1-planar graph with all 4 kite edges for every crossing. Then G has a B_{16}-VPG 4-string representation.*

We postpone the proof Theorem 7.11 to the end of this chapter as we need to build some preliminary notion, and also wish to take a detour in order point out a relationship to the results presented in Chapter 4.

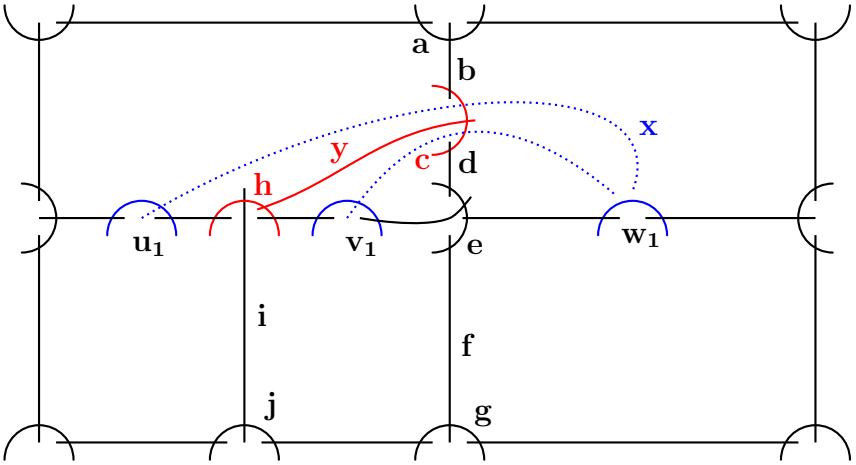
A *1-visibility* representation of a graph displays each vertex as a horizontal vertex-segment, called a *bar*, and each edge as a vertical edge-segment between the segments of the vertices, such that each edge-segment crosses at most one vertex-segment and each vertex-segment is crossed by at most one edge-segment. See also Figure 7.10. A graph is *1-visible* if it has such a representation.

Lemma 7.12 (F. Brandenburg [22])**.** *If G is a 1-planar graph with a 1-planar embedding that has all four kite edges for every crossing, then G has a 1-visibility representation where every crossing is represented as shown in Figure 7.10.*

Recall the definition of a single-vertical representation of a graph from Chapter 4: A *single-vertical object* is a connected set $S \subset \mathbb{R}^2$ of the form $S = s_0 \cup s_1 \cup \cdots \cup s_k$, where s_0 is a vertical segment and $s_1, \ldots, s_k$ are horizontal segments, for some finite k. A *single-vertical representation* is an intersection representation of such objects.



(a) A 1-planar embedding of the graph.



(b) An illustration of its string representation.

Figure 7.9: A 1-planar graph that does not have a 1-string representation.

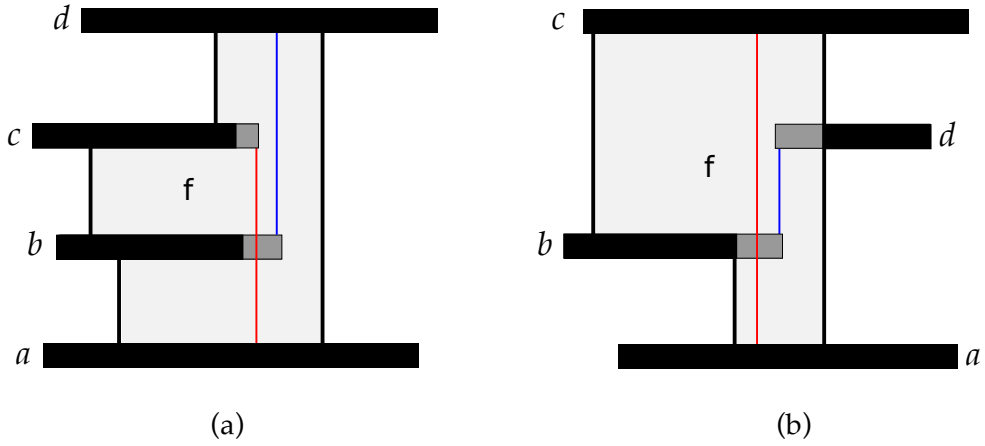


Figure 7.10: The two ways of representing a crossing or edges (a, c) (the visibility line is shown in red) and (b, d) (the visibility line is shown in blue) in a 1-visibility representation of a 1-planar graph. Figure based on [22].

We can show the following:

Theorem 7.13. *Let G be a 1-planar graph with all 4 kite edges for every crossing. Then G has a single-vertical representation such that every vertex object consists of a single vertical segment and at most 4 horizontal segments.*

Proof. The edges of every 1-planar graphs can be partitioned into a forest and a planar graph [1]. The edges of the planar graph be further partitioned into 3 forests [37]. Thus, every 1-planar graph can be partitioned into 4 forests and we can orient the edges so that every vertex is incident to at most 4 incoming edges. Assume that we have fixed such an orientation of G. Construct a 1-visibility representation of G using Lemma 7.12. For every vertex bar b_v, we can add $k \leq 4$ vertical segments that connect it to the bars of its k incoming neighbours in order to create all incoming edges as shown in Figure 7.11. Every vertical segment is positioned along the visibility line that connects the two bars. The horizontal segment is positioned in the center of the bar. As all the kite edges exist, all the created intersections are allowed. Rotating the representation by 90°, we obtain a single-vertical representation of G. □

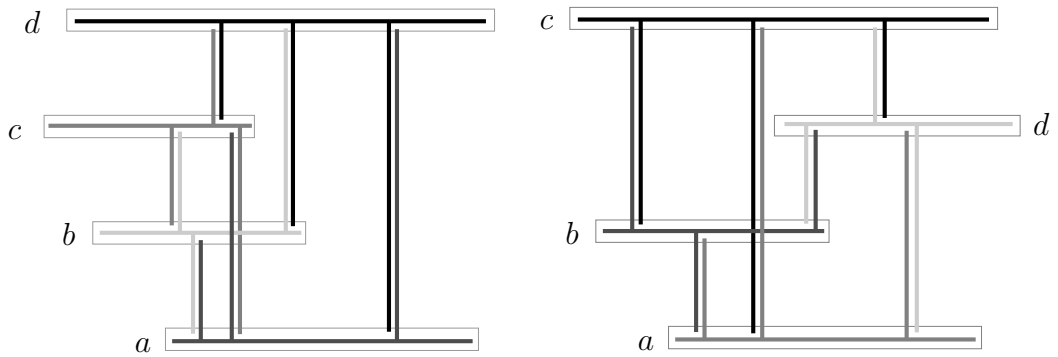


Figure 7.11: The possible locations of vertical segments connecting the bars. The segments are added based on the orientations of edges.

We can now also use the single-vertical representations to provide a B_k-VPG 4-string representation:

Lemma 7.14. *If G has a single-vertical representation where objects have at most k horizontal segments that end on the vertical line (i.e., attach but do not cross it), then G is a B_{4k}-VPG graph.*

Proof. Assume that a single-vertical representation R with at most k horizontal segments in each object is given.

Then, for every vertex v, trace around the corresponding object S_v with a curve $\mathbf{v}$. See Figure 7.12 for an illustration. The vertical segment of S_v will become part of $\mathbf{v}$. For a horizontal segment s, we bend $\mathbf{v}$ rightward or leftward, follow s to its very end (this is a point of contact with another curve), create a proper intersection with the other curve, add two bends and lead $\mathbf{v}$ along s back and to the next horizontal segment.

As tracing each vertical segment adds 4 bends and there are at most 4 of them, we get at most 16 bends for each curve. □

Now we can prove Theorem 7.11:

Proof of Theorem 7.11. If we start with the single-vertical representation of Theorem 7.13, then the objects of two vertices v and w intersect at most 3 times: once for edge (v, w), once

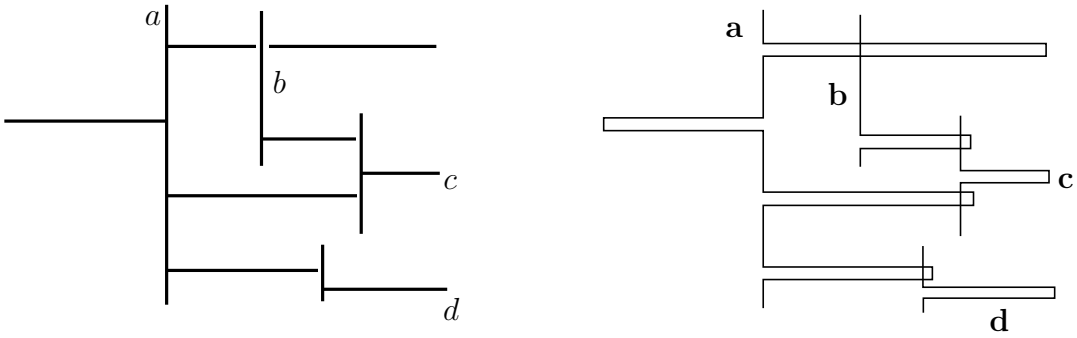


Figure 7.12: Tracing around objects in a single-vertical representation to produce a string representation.

(perhaps) for some edge (x, v) whose segment intersects the bar of w, and once (perhaps) for some edge (y, w) whose segment intersects the bar of v. The latter two can exist only if (v, w) was a kite edge, hence the vertical bar did not cross vertex-segments. We can then remove the intersection for edge (v, w) by removing that vertical bar, yielding a single-vertical representation where any two objects intersect at most twice. Any such intersection yields two crossings of the corresponding strings, hence we obtain a 4-string representation. □

Chapter 8

Future Directions

In this work, we investigated string representations with a focus on restricting the shapes of the curves, number of bends, number or intersections between two curves, and order of intersections along curves. In this chapter, we point out questions that remain open, and suggest directions for the future work in this field.

8.1 Graphs with no string representations

We know that a full subdivision of K_5 and $K_{3,3}$ are not string graphs (and neither is any full subdivision of any non-planar graph). As string graphs are closed under taking induced minors, any graph G that contains an induced minor H that is not a string graph is not a string graph either. However, are those the only graphs that are not string graphs? No other examples of graphs that are not string graphs are known.

We think that if a graph does not contain a full subdivision of a non-planar graph as an induced minor, then it is a string graph. Note however that given a certificate, such a property can be verified in polynomial time. This would situate the recognition problem of string graphs in co-NP. However, as the problem is known to be NP-complete, which would imply that NP = co-NP. Thus, this question is especially interesting from the theoretical perspective.

One can quickly "fix" 1-subdivisions of K_5 and $K_{3,3}$ to become string graphs by adding an edge that connects a single pair of the subdivision vertices. Such an edge addition will produce a string graph. This can be argued using our results about 1-planar graphs: such a graph has a 1-planar embedding in which the added edge is a kite edge for the crossing (see Theorem 7.9). So, a related question is:

Question 8.1. *Given a graph G that is not a string graph, what is the minimum subset of edges that needs to be added to G in order to obtain a supergraph that is a string graph? Can one characterize such supergraphs of graphs without string representation?*

This question was proposed before by Bokal et al. in [19] who called this the *string crossing number*.

8.2 B_1-VPG and segment representations

Many research questions investigated in this thesis were motivated by the question whether planar graphs have $\{\llcorner,\ulcorner\}$-representations. For instance, in Chapter 3, we proved that planar graphs have representations that are simultaneously 1-string and B_2-VPG, and in Chapter 5, we presented constructions for B_1-VPG and $\{\llcorner,\ulcorner\}$-representations of some subclasses of planar graphs. Shortly after submitting this thesis for public display, Gonçalves et al. [55] proved that planar graphs have $\{\llcorner\}$-representations. The technique in their paper uses concepts similar to those in Chapter 3, i.e., splitting triangulated disks along edges, constructing representations with prescribed layouts inductively and merging them together. However, the actual way of splitting the graph is novel. Their result also provides a different proof of Scheinerman's conjecture due to the transformation given by Middendorf and Pfeiffer [74] (see Section 6.3.4).

We consider the fact that $\llcorner$-shapes are sufficient and the use of $\ulcorner$-shapes is unnecessary to represent all planar graphs very surprising. This brings up the question of what is the difference between the classes of graphs with $\{\llcorner\}$- and $\{\llcorner,\ulcorner\}$-representations. It seems believable that the two classes differ, however, we do not know of an example of a graph with $\{\llcorner,\ulcorner\}$-representation that would require both the shapes. The technique of Chaplick

et al. [33] presented in Section 5.3 is not applicable as a single ⌞ intersecting a ⌜ cannot separate a closed area of the plane.

Another research direction follows up on the fact that B_1-VPG representations consisting of curves with shapes ⌞ and ⌜ only can be transformed into segment representations (see Section 6.3.4). One might wonder whether the converse holds, i.e., whether any graph in Seg has an {⌞,⌜}-representation. This is false, and in fact a stronger result holds:

Lemma 8.2 (Chaplick et al. [33])**.** *For every $k \geq 1$, B_k-VPG is not a subset of* Seg*.*

For $k = 2$, an example of a graph that is not in Seg is the graph in Figure 5.15. However, assume that we are given a graph that is known to be simultaneously in B_1-VPG and Seg. Given a segment representation, is it possible to produce a B_1-VPG representation?

We can generalize this question to all B_k-VPG graphs. Lemma 8.2 stipulates that there is no inclusion between Seg and B_k-VPG for any k. Recognition of both B_k-VPG and Seg is NP-hard, but recognition of Seg is hard in the existential theory of the reals (so-called $\exists\mathbb{R}$) [69] and thus appears to be harder. What about the recognition problem of Seg when a B_k-VPG representation is given?

Conjecture 8.3. *For any fixed k, testing whether a B_k-VPG graph G is in* Seg *is hard in the existential theory of the reals even if a B_k-VPG representation is given.*

Lastly, Felsner et al. [42] showed that line graphs of planar graphs have B_1-VPG representations. They also showed that complements of planar graphs, so called *co-planar* graphs, have B_{19}-VPG representations. However, it is not known whether 19 bends are necessary, and this seems unlikely. The question whether every co-planar graph belongs to Seg is open as well.

8.3 Outer-string graphs

The class of outer-string graphs proved useful from the algorithmic perspective in that some hard problems become tractable on outer-string graphs. In Chapter 4, we showed how to

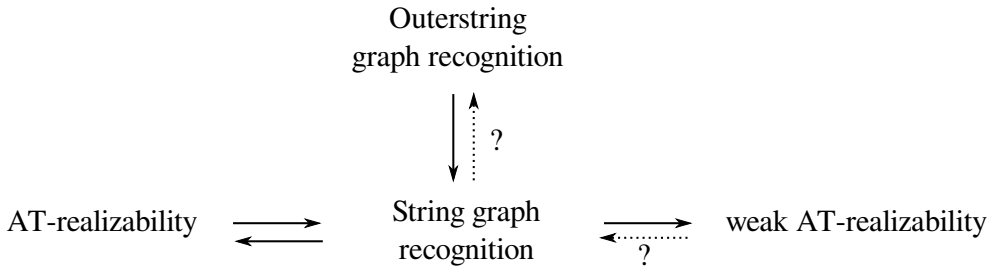


Figure 8.1: Known reductions between variants of At-Realizability variants and string graph recognition.

use such algorithms for solving problems on outer-string in order to produce approximation algorithms for single-vertical and B_2-VPG graphs. The class of single-vertical graphs is rich and includes B_1-VPG graphs, planar graphs, 1-planar graph—see Theorem 7.13. This shows that algorithmic results of any kind for outer-string graphs have potential to have deep consequences, and thus are very interesting.

One of the interesting questions that concerns the class of the outer-string graphs itself is the recognition. In Section 8.3, we provided a rather simple argument that the recognition is in NP. However, the question whether outer-string graphs can be recognized in polynomial time is open. Based on our private communication with J. Kratochvíl (the author of one NP-hardness proofs for string graph recognition [67]) there is not a clear opinion about this question. While Kratochvíl thinks that recognizing outer-string graphs should be NP-hard, Middendorf and Pfeiffer (the authors of the other proof presented in [74]) apparently did not share his opinion and felt that one should be able to describe outer-string graphs using some forbidden obstructions.

The problem of string graph recognition is closely related to At-Realizability and its weak variant. It is known that At-Realizability reduces to the recognition problem of string graphs [68], and vice versa. Recognizing string graphs can also be reduced to weak At-Realizability, i.e., the version of At-Realizability where edges that are allowed to intersect do not have to intersect. This is the crucial ingredient in arguing that the recognition problem of string graphs is in NP, because weak At-Realizability is in

NP [78]. But, does weak At-Realizability reduce to recognizing string graphs?[1]

We have shown that recognizing outer-string graphs reduces to recognizing string graphs. However, we do not know if the opposite is true. Note that this would imply NP-hardness of outer-string recognition. Furthermore, we are not aware of direct reductions between At-Realizability and the weak version of the problem. Alternatively, can we prove a version of At-Realizability NP-hard for which the reduction of Lemma 2.7 leads to outer-string graphs?

8.4 1-planar graphs

To the best of our knowledge, there is no previous work on string representations of 1-planar graphs. We showed that there are 1-planar graphs that are not string graphs, and furthermore, that if a planar graph has at least 1 kite edge for every crossing, it is a string graph (see Theorem 7.11). However, we find it very likely that there are some string graphs that do not have a kite for every crossing, and yet, they have string representations.

Conjecture 8.4. *There is a 1-planar string graph G such that G has no 1-planar embedding in which every crossing is drawn with a kite edge.*

More generally, we are interested in a characterization of kite edges that are truly required for a string representation to exist.

We also showed that 1-planar graphs with at least 1 kite edge for every crossing have 4-string representations. We further provided an example of such a 1-planar graph that does not have 1-string representation, but has a 2-string representation. Despite our efforts, we were unable to produce a 1-planar graph that is a string graph and requires more than 2 intersections between two curves.

Conjecture 8.5. *Every 1-planar string graph has a 2-string representation.*

[1] It does, because weak At-Realizability is in NP while string graph recognition is NP-hard. But, is there a simple, direct reduction, similar to the one in Lemma 2.7?

Furthermore, while the existence of a single kite edge for each crossing is sufficient to show that a 1-planar graph is a string graph, we currently have no bound on the number of bends needed for a B_k-VPG representation. Thus, it is an open question whether there are 1-planar string graphs with kite edges where the number of bends needed is unbounded. We find it very unlikely and conjecture the following:

Conjecture 8.6. *There is a constant k such that every 1-planar graph with at least 1 kite edge for every crossing has a B_k-VPG representation.*

We also proved that every 1-planar graph that has all kite edges for every crossing has a 4-string B_{16}-VPG representation. We do not have any evidence that either of the two, 16 bends or 4 crossings, is actually required if all kite edges exist. In fact, this appears to be quite a large gap between planar graphs that are known to have representations that are simultaneously 1-string and B_2-VPG (see Theorem 3.1). Thus, we conjecture the following:

Conjecture 8.7. *Every 1-planar graph with all kite edges for every crossing has a 2-string B_k-VPG representation with $k < 16$.*

As far as the complexity is concerned, testing whether a 1-planar graph is NP-hard. One can provide a short argument by reduction from a string graph recognition. Given a graph G with m edges, one can subdivide every edge with $m - 1$ vertices. The resulting graph G' has a 1-planar embedding and is a string graph if and only if G is a string graph, and G' is only polynomially larger than G. Thus, testing if G' is a string graph cannot be easier than testing if G is a string graph. The hardness of string graph recognition was proved by Kratochvíl [68] by reduction from ∃-Realizability. It would be interesting to know if there is a 1-planar variant of ∃-Realizability that would be NP-hard as well. We do suspect that the graphs used in the hardness reduction by Kratochvíl in [68] can be made 1-planar by subdividing the edges with additional vertices.

Lastly, there are a number of subclasses of 1-planar graphs and classes that related to them and were not investigated here. Examples of such graph classes are fan-planar graphs [60] or more generally, k-planar graphs. What can we show about their string representations?

8.5 Order-preserving string representations

In Chapter 6, we presented three notions of order-preserving string representations: linear, cyclic and selective. We presented both positive and negative results. The future work in this field should explore more graph classes to see whether order-preserving representations (in one of these models) exist or not. For instance, we showed that series-parallel graphs have cyclic and selective order-preserving representations with respect to some planar embedding. However, can they have such a representation with *any* planar embedding?

Secondly, following up on the complexity questions discussed in Section 8.3 what is the complexity of testing whether an order-preserving 1-string representation exists? Given the NP-hardness of the abstract graph realization problem [68, 75], this is very likely NP-hard if we are allowed to prescribe an arbitrary cyclic ordering of edges around each vertex (i.e., not from a planar drawing). But is it NP-hard for plane graphs?

References

[1] Eyal Ackerman. A note on 1-planar graphs. *Discrete Appl. Math.*, 175:104–108, 2014.
Cited on page(s): 154

[2] Pankaj K. Agarwal, Marc van Kreveld, and Subhash Suri. Label placement by maximum independent set in rectangles. *Computational Geometry*, 11(3):209 – 218, 1998.
Cited on page(s): 67

[3] Abu Reyan Ahmed, Felice De Luca, Sabin Devkota, Alon Efrat, Md. Iqbal Hossain, Stephen G. Kobourov, Jixian Li, Sammi Abida Salma, and Eric Welch. L-Graphs and Monotone L-Graphs. *CoRR*, abs/1703.01544, 2017.
Cited on page(s): 92, 93, 100

[4] Oswin Aichholzer, Florian Ebenführe, Irene Parada, Alexander Pilz, and Birgit Vogtenhuber. On semi-simple drawings of the complete graph. *XVII Spanish Meeting on Computational Geometry*, 2017, to appear.
Cited on page(s): 2

[5] Muhammad Jawaherul Alam, Therese Biedl, Stefan Felsner, Andreas Gerasch, Michael Kaufmann, and Stephen G. Kobourov. Linear-time algorithms for hole-free rectilinear proportional contact graph representations. *Algorithmica*, 67(1):3–22, 2013.
Cited on page(s): 165

[6] Muhammad Jawaherul Alam, Therese Biedl, Stefan Felsner, Michael Kaufmann, and Stephen Kobourov. Proportional contact representations of planar graphs. http://page.math.tu-berlin.de/~felsner/Paper/prop_contact.pdf, 2012. Full version

appeared in [5], but it excludes the result referenced in Section 5.1.3.
Cited on page(s): 91

[7] Takao Asano, Shunji Kikuchi, and Nobuji Saito. A linear algorithm for finding Hamiltonian cycles in 4-connected maximal planar graphs. *Discr. Applied Mathematics*, 7(1):1 – 15, 1984.
Cited on page(s): 24, 25, 49

[8] Andrei Asinowski, Elad Cohen, Martin Charles Golumbic, Vincent Limouzy, Marina Lipshteyn, and Michal Stern. String graphs of k-bend paths on a grid. *Electronic Notes in Discrete Mathematics*, 37:141–146, 2011.
Cited on page(s): 13

[9] Christopher Auer, Franz J. Brandenburg, Andreas Gleißner, and Josef Reislhuber. 1-planarity of graphs with a rotation system. *J. Graph Algorithms Appl.*, 19(1):67–86, 2015.
Cited on page(s): 150

[10] Seymour Benzer. On the topology of the genetic fine structure. *Proceedings of the National Academy of Sciences of the United States of America*, 45(11):1607–1620, 1959.
Cited on page(s): 8

[11] Therese Biedl and Martin Derka. 1-string B_1-VPG Representations of Planar Partial 3-Trees and Some Subclasses. In *Proceedings of the 27th Canadian Conference on Computational Geometry, CCCG 2015*, 2015.
Cited on page(s): 85, 105

[12] Therese Biedl and Martin Derka. 1-String B_2-VPG Representation of Planar Graphs. In *31st International Symposium on Computational Geometry, SoCG 2015*, pages 141–155. LiPiCS, 2015.
Cited on page(s): 24, 94

[13] Therese Biedl and Martin Derka. 1-String B_2-VPG Representation of Planar Graphs. *Journal of Computational Geometry*, 7(2):191–215, 2016.
Cited on page(s): 24

[14] Therese Biedl and Martin Derka. Order-preserving 1-string representations of planar graphs. In *SOFSEM 2017: Theory and Practice of Computer Science - 43rd International Conference on Current Trends in Theory and Practice of Computer Science, Proceedings*, volume 10139 of *Lecture Notes in Computer Science*, pages 283–294. Springer, 2017.
Cited on page(s): 114

[15] Therese Biedl and Martin Derka. Splitting B_2-VPG Graphs into Outer-string and Co-comparability Graphs. In *Algorithms and Data Structures Symposium, WADS 2017*, to appear, 2017.
Cited on page(s): 68

[16] Therese Biedl, Giuseppe Liotta, and Fabrizio Montecchiani. On visibility representations of non-planar graphs. In *32nd International Symposium on Computational Geometry, SoCG 2016*, volume 51 of *LIPIcs*, pages 19:1–19:16. Schloss Dagstuhl - Leibniz-Zentrum fuer Informatik, 2016.
Cited on page(s): 150

[17] Therese Biedl and Lesvia Elena Ruiz Velázquez. Drawing planar 3-trees with given face areas. *Comput. Geom.*, 46(3):276–285, 2013.
Cited on page(s): 94

[18] Hans Bodlaender. Planar graphs with bounded treewidth. Technical Report RUU-CS-88-14, Rijksuniversiteit Utrecht, 1988.
Cited on page(s): 104

[19] Drago Bokal, Éva Czabarka, László A. Székely, and Imrich Vrt'o. General lower bounds for the minor crossing number of graphs. *Discrete & Computational Geometry*, 44(2):463–483, 2010.
Cited on page(s): 158

[20] Flavia Bonomo, María Pía Mazzoleni, and Maya Stein. Clique coloring B_1-EPG graphs. *Discrete Mathematics*, 340(5):1008–1011, 2017.
Cited on page(s): 21

[21] Marin Bougeret, Stéphane Bessy, Daniel Gonçalves, and Christophe Paul. On independent set on B1-EPG graphs. In *Approximation and Online Algorithms - 13th International Workshop, WAOA 2015, Revised Selected Papers*, volume 9499 of *Lecture Notes in Computer Science*, pages 158–169. Springer, 2015.
Cited on page(s): 21

[22] Franz J. Brandenburg. 1-visibility representations of 1-planar graphs. *J. Graph Algorithms Appl.*, 18(3):421–438, 2014.
Cited on page(s): 150, 152, 154

[23] Andreas Brandstädt, Van Bang Le, and Jeremy P. Spinrad. *Graph Classes: A Survey*. Society for Industrial and Applied Mathematics, Philadelphia, PA, USA, 1999.
Cited on page(s): 88, 91

[24] Sergio Cabello, Jean Cardinal, and Stefan Langerman. The clique problem in ray intersection graphs. *Discrete & Computational Geometry*, 50(3):771–783, 2013.
Cited on page(s): 83

[25] Sergio Cabello and Miha Jejčič. Refining the hierarchies of classes of geometric intersection graphs. *Electronic Notes in Discrete Mathematics*, 54:223–228, 2016.
Cited on page(s): 16

[26] Jean Cardinal, Stefan Felsner, Tillmann Miltzow, Casey Tompkins, and Birgit Vogtenhuber. Intersection graphs of rays and grounded segments. Technical Report 1612.03638 [cs.DM], ArXiV, 2016.
Cited on page(s): 75

[27] Jean Cardinal, Stefan Felsner, Tillmann Miltzow, Casey Tompkins, and Birgit Vogtenhuber. Intersection graphs of rays and grounded segments. In *Graph-Theoretic Concepts in Computer Science WG 2017*, to appear, 2017.
Cited on page(s): 75

[28] Daniele Catanzaro, Steven Chaplick, Stefan Felsner, Bjarni V. Halldórsson, Magnús M. Halldórsson, Thomas Hixon, and Juraj Stacho. Max point-tolerance graphs. *CoRR*, abs/1508.03810, 2015.
Cited on page(s): 93

[29] Jérémie Chalopin and Daniel Gonçalves. Every planar graph is the intersection graph of segments in the plane: extended abstract. In *ACM Symposium on Theory of Computing, STOC 2009*, pages 631–638. ACM, 2009.
Cited on page(s): 12, 15

[30] Jérémie Chalopin, Daniel Gonçalves, and Pascal Ochem. Planar graphs are in 1-string. In *ACM-SIAM Symposium on Discrete Algorithms*, SODA '07, pages 609–617. SIAM, 2007.
Cited on page(s): 12, 23, 24

[31] Jérémie Chalopin, Daniel Gonçalves, and Pascal Ochem. Planar graphs have 1-string representations. *Discrete & Computational Geometry*, 43(3):626–647, 2010.
Cited on page(s): 12, 23, 24, 25, 31, 56, 57, 94, 123

[32] Steven Chaplick, Stefan Felsner, Udo Hoffmann, and Veit Wiechert. Grid intersection graphs and order dimension. *CoRR*, abs/1512.02482, 2015.
Cited on page(s): 93

[33] Steven Chaplick, Vít Jelínek, Jan Kratochvíl, and Tomáš Vyskočil. Bend-Bounded Path Intersection Graphs: Sausages, Noodles, and Waffles on a Grill. In *Graph-Theoretic Concepts in Computer Science - 38th International Workshop, WG 2012, Revised Selected Papers*, volume 7551 of *Lecture Notes in Computer Science*, pages 274–285. Springer, 2012.
Cited on page(s): 19, 109, 111, 159

[34] Steven Chaplick, Stephen G. Kobourov, and Torsten Ueckerdt. Equilateral L-Contact Graphs. In *Graph-Theoretic Concepts in Computer Science - 39th International Workshop, WG 2013, Revised Papers*, volume 8165 of *Lecture Notes in Computer Science*, pages 139–151. Springer, 2013.
Cited on page(s): 88, 91

[35] Steven Chaplick and Torsten Ueckerdt. Planar graphs as VPG-graphs. *J. Graph Algorithms Appl.*, 17(4):475–494, 2013.
Cited on page(s): 13, 23

[36] Natalia de Castro, Francisco Javier Cobos, Juan Carlos Dana, Alberto Márquez, and Marc Noy. Triangle-free planar graphs and segment intersection graphs. *J. Graph Algorithms Appl.*, 6(1):7–26, 2002.
Cited on page(s): 12

[37] Hubert de Fraysseix and Patrice Ossona de Mendez. Regular orientations, arboricity, and augmentation. In *Graph Drawing, DIMACS International Workshop, GD '94*, volume 894 of *Lecture Notes in Computer Science*, pages 111–118. Springer, 1994.
Cited on page(s): 154

[38] Hubert de Fraysseix, Patrice Ossona de Mendez, and János Pach. Representation of planar graphs by segments. *Intuitive Geometry*, 63:109–117, 1991.
Cited on page(s): 12, 13, 86, 136

[39] Hubert de Fraysseix, Patrice Ossona de Mendez, and Pierre Rosenstiehl. On triangle contact graphs. *Combinatorics, Probability & Computing*, 3:233–246, 1994.
Cited on page(s): 136

[40] Gideon Ehrlich, Shimon Even, and Robert Endre Tarjan. Intersection graphs of curves in the plane. *J. Comb. Theory, Ser. B*, 21(1):8–20, 1976.
Cited on page(s): 8, 10, 137

[41] Ehab S. El-Mallah and Charles J. Colbourn. Partitioning the edges of a planar graph into two partial k-trees. *Congressus Numerantium 66*, pages 69 – 80, 1988.
Cited on page(s): 98

[42] Stefan Felsner, Kolja B. Knauer, George B. Mertzios, and Torsten Ueckerdt. Intersection graphs of L-shapes and segments in the plane. In *Mathematical Foundations of Computer Science (MFCS'14), Part II*, volume 8635 of *Lecture Notes in Computer Science*, pages 299–310. Springer, 2014.
Cited on page(s): 31, 91, 92, 93, 94, 121, 123, 159

[43] Holger Flier, Matús Mihalák, Peter Widmayer, Anna Zych, Yusuke Kobayashi, and Anita Schöbel. Selecting vertex disjoint paths in plane graphs. *Networks*, 66(2):136–144, 2015.
Cited on page(s): 16

[44] Jacob Fox and János Pach. Separator theorems and Turán-type results for planar intersection graphs. *Adv. Math.*, 219:1070–1080, 2008.
Cited on page(s): 20

[45] Jacob Fox and János Pach. A separator theorem for string graphs and its applications. *Combinatorics, Probability & Computing*, 19(3):371–390, 2010.
Cited on page(s): 20

[46] Jacob Fox and János Pach. Computing the independence number of intersection graphs. In *Proceedings of the Twenty-Second Annual ACM-SIAM Symposium on Discrete Algorithms, SODA 2011*, pages 1161–1165, 2011.
Cited on page(s): 20, 82

[47] Jacob Fox and János Pach. Coloring K_k-free intersection graphs of geometric objects in the plane. *Eur. J. Comb.*, 33(5):853–866, 2012.
Cited on page(s): 15

[48] Jacob Fox and János Pach. String graphs and incomparability graphs. In *Symposuim on Computational Geometry 2012, SoCG '12*, pages 405–414. ACM, 2012.
Cited on page(s): 15

[49] Jacob Fox and János Pach. Applications of a new separator theorem for string graphs. *Combinatorics, Probability & Computing*, 23(1):66–74, 2014.
Cited on page(s): 20

[50] Mathew C. Francis and Abhiruk Lahiri. VPG and EPG bend-numbers of Halin graphs. *Discrete Applied Mathematics*, 215:95–105, 2016.
Cited on page(s): 105, 107

[51] Tomás Gavenčiak, Przemyslaw Gordinowicz, Vít Jelínek, Pavel Klavík, and Jan Kratochvíl. Cops and robbers on string graphs. In *Algorithms and Computation - 26th International Symposium, ISAAC 2015, Proceedings*, volume 9472 of *Lecture Notes in Computer Science*, pages 355–366. Springer, 2015.
Cited on page(s): 20

[52] Emilio Di Giacomo, Giuseppe Liotta, and Fabrizio Montecchiani. Drawing outer 1-planar graphs with few slopes. In *22nd International Symposium on Graph Drawing, GD 2014, Revised Selected Papers*, volume 8871 of *Lecture Notes in Computer Science*, pages 174–185. Springer, 2014.
Cited on page(s): 147

[53] Martin Charles Golumbic. *Algorithmic Graph Theory and Perfect Graphs*. Academic Press, New York, 1st edition, 1980.
Cited on page(s): 83

[54] Martin Charles Golumbic, Doron Rotem, and Jorge Urrutia. Comparability graphs and intersection graphs. *Discrete Mathematics*, 43(1):37–46, 1983.
Cited on page(s): 74

[55] Daniel Gonçalves, Lucas Isenmann, and Claire Pennarun. Planar graphs as L-intersection or L-contact graphs. *CoRR*, abs/1706.10047, 2017.
Cited on page(s): 15, 21, 66, 85, 130, 158

[56] Rajeev Govindan, Michael A. Langston, and Xudong Yan. Approximating the path-width of outerplanar graphs. *Inf. Process. Lett.*, 68(1):17–23, 1998.
Cited on page(s): 121

[57] Rudof Halin. Studies on minimally n-connected graphs. In *Combinatorial Mathematics and its Applications*, pages 129–136. Academic Press, London, 1971.
Cited on page(s): 104

[58] Sariel Har-Peled and Kent Quanrud. Approximation algorithms for polynomial-expansion and low-density graphs. In *23rd Annual European Symposium on Algorithms ESA 2015*, volume 9294 of *Lecture Notes in Computer Science*, pages 717–728. Springer,

2015.
Cited on page(s): 20

[59] Irith Ben-Arroyo Hartman, Ilan Newman, and Ran Ziv. On grid intersection graphs. *Discrete Mathematics*, 87(1):41–52, 1991.
Cited on page(s): 12, 13

[60] Michael Kaufmann and Torsten Ueckerdt. The density of fan-planar graphs. *CoRR*, abs/1403.6184, 2014.
Cited on page(s): 162

[61] J. Mark Keil, Joseph S. B. Mitchell, Dinabandhu Pradhan, and Martin Vatshelle. An algorithm for the maximum weight independent set problem on outerstring graphs. *Comput. Geom.*, 60:19–25, 2017.
Cited on page(s): 20, 81, 139

[62] J. Mark Keil and Lorna Stewart. Approximating the minimum clique cover and other hard problems in subtree filament graphs. *Discrete Applied Mathematics*, 154(14):1983–1995, 2006.
Cited on page(s): 68, 81, 83

[63] Mark Keil, Joseph S. B. Mitchell, Dinabandhu Pradhan, and Martin Vatshelle. An algorithm for the maximum weight independent set problem on outersting graphs. In *Proceedings of the 27th Canadian Conference on Computational Geometry, CCCG 2015*, 2015.
Cited on page(s): 20, 81, 139

[64] Stephen G. Kobourov, Giuseppe Liotta, and Fabrizio Montecchiani. An annotated bibliography on 1-planarity. *CoRR*, abs/1703.02261, 2017.
Cited on page(s): 150

[65] Stephen G. Kobourov, Torsten Ueckerdt, and Kevin Verbeek. Combinatorial and geometric properties of planar Laman graphs. *SIAM Symposium on Discrete Algorithms (SODA 2013)*, pages 1668–1678, 2013.
Cited on page(s): 89, 90, 136

[66] Paul Koebe. Kontaktprobleme auf der konformen Abbildung. *Ber. Verh. Sächs. Akad. Wiss. Leipzig, Math.-Phys. Kl.*, 88:141–164, 1936.
Cited on page(s): 8

[67] Jan Kratochvíl. String graphs I. The number of critical nonstring graphs is infinite. *J. Comb. Theory, Ser. B*, 52(1):53–66, 1991.
Cited on page(s): 9, 10, 14, 15, 141, 160

[68] Jan Kratochvíl. String graphs II. Recognizing string graphs is NP-hard. *J. Comb. Theory, Ser. B*, 52(1):67–78, 1991.
Cited on page(s): 16, 17, 141, 142, 143, 160, 162, 163

[69] Jan Kratochvíl. A special planar satisfiability problem and a consequence of its NP-completeness. *Discrete Applied Mathematics*, 52(3):233–252, 1994.
Cited on page(s): 17, 19, 134, 159

[70] Jan Kratochvíl and Jiří Matoušek. String graphs requiring exponential representations. *J. Comb. Theory, Ser. B*, 53(1):1–4, 1991.
Cited on page(s): 17, 109, 137, 138

[71] Abhiruk Lahiri, Joydeep Mukherjee, and C. R. Subramanian. Maximum independent set on B_1-VPG graphs. In *Proceedings of Combinatorial Optimization and Applications COCOA 2015, Lecture Notes in Computer Science 9486*, pages 633–646. Springer, 2015.
Cited on page(s): 21, 67, 68, 69, 80

[72] James R. Lee. Separators in region intersection graphs. In *Innovations in Theoretical Computer Science, ITCS'17*, 2017.
Cited on page(s): 20

[73] Jiří Matoušek. Near-optimal separators in string graphs. *CoRR*, abs/1302.6482, 2013.
Cited on page(s): 20

[74] Matthias Middendorf and Frank Pfeiffer. The max clique problem in classes of string-graphs. *Discrete Mathematics*, 108(1-3):365–372, 1992.
Cited on page(s): 14, 20, 92, 132, 133, 158, 160

[75] Matthias Middendorf and Frank Pfeiffer. Weakly transitive orientations, Hasse diagrams and string graphs. *Discrete Mathematics*, 111(1-3):393–400, 1993.
Cited on page(s): 163

[76] Alexandre Rok and Bartosz Walczak. Outerstring graphs are χ-bounded. In *30th Annual Symposium on Computational Geometry, SOCG'14*, pages 136–143. ACM, 2014.
Cited on page(s): 20

[77] Marcus Schaefer. Complexity of some geometric and topological problems. In *Graph Drawing, 17th International Symposium, GD 2009, Chicago, IL, USA, September 22-25, 2009. Revised Papers*, pages 334–344, 2009.
Cited on page(s): 19

[78] Marcus Schaefer, Eric Sedgwick, and Daniel Štefankovič. Recognizing string graphs is in NP. *Journal of Computer and System Sciences*, 67(2):365–380, 2003.
Cited on page(s): 17, 19, 142, 161

[79] Edward R. Scheinerman. *Intersection Classes and Multiple Intersection Parameters of Graphs*. PhD thesis, Princeton University, 1984.
Cited on page(s): 11

[80] F. W. Sinden. Topology of thin film rc-circuits. *Bell System Technical Journal*, 45:1639–1662, 1966.
Cited on page(s): 8

[81] Carsten Thomassen. Rectilinear drawings of graphs. *Journal of Graph Theory*, 12(3):335–341, 1988.
Cited on page(s): 147

[82] William Thomas Tutte. Convex representations of graphs. *Proc. London Math. Soc*, 10(38):304–320, 1960.
Cited on page(s): 25

[83] William Thomas Tutte. How to draw a graph. *Proc. London Math. Soc*, 13(52):743–768, 1963.
Cited on page(s): 25

[84] W. Wessel and R. Pöschel. On circle graphs. In Horst Sachs, editor, *Graphs, Hypergraphs and Applications*, volume 73 of *Teubner-Texte zur Mathematik*, pages 207–210. Teubner, 1985.
Cited on page(s): 121

[85] Hassler Whitney. A theorem on graphs. *The Annals of Mathematics*, 32(2):387–390, 1931.
Cited on page(s): 24

[86] Hassler Whitney. Congruent graphs and the connectivity of graphs. *American Journal of Mathematics*, 54(1):150–168, 1932.
Cited on page(s): 152