Roma Tre University
Ph.D. in Computer Science and Engineering

Planar Graphs with Vertices in Prescribed Regions: models, algorithms, and complexity

Giordano Da Lozzo

Planar Graphs with Vertices in Prescribed Regions: models, algorithms, and complexity

A thesis presented by
Giordano Da Lozzo
in partial fulfillment of the requirements for the degree of
Doctor of Philosophy
in Computer Science and Engineering

Roma Tre University
Department of Engineering

Spring 2015

COMMITTEE:
Prof. Giuseppe Di Battista
Prof. Maurizio Patrignani

REVIEWERS:
Prof. Seok-Hee Hong
Prof. Michael Kaufmann

a mia nonna Anna

Imagination is more important than knowledge. For knowledge is limited to all we now know and understand, while imagination embraces the entire world, and all there ever will be to know and understand. – Albert Einstein

Acknowledgments

My best acknowledgments go to my advisors, Giuseppe Di Battista and Maurizio Patrignani. They introduced me to the beautiful topics which they actively research and instilled in me the deep passion they share in their work. Ever since I met them their love for science and their restless curiosity have been a constant source of inspiration for me. Working with them has been a great honor and one of the most important events of my life, as it led me to answer the question "What do I want to do when I grow up?".

A special thank goes to Patrizio Angelini. Over the course of this thesis he has been a third advisor, an amazing research partner, and – most of all – a true friend. I am tremendously indebted to him for all I have gained from our fascinating discussions.

I would like to thank Frabrizio Frati and Ignaz Rutter for the opportunity they gave me to work alongside them. I am glad that this thesis is just the start of our collaboration.

I would like to thank Jan Kratochvìl and Dorothea Wagner for allowing me to spend very pleasant and productive time as a visiting doctoral student at the Charles University and the Karlsruhe Institute of Technology, respectively.

I would like to thank Seok-Hee Hong and Michael Kaufmann for carefully reviewing this thesis.

I would like to thank all my coauthors for the great time I had while working with each of them: Patrizio Angelini, Carla Binucci, Davide Cannone, Walter Didimo, Marco Di Bartolomeo, Giuseppe Di Battista, Fabrizio Frati, Luca Grilli, Seok-Hee Hong, Francesco Ingrassia, Vìt Jelìnek, Jan Kratochvìl, Anna Lubiw, Fabrizio Montecchiani, Daniel Neuwirth, Maurizio Patrignani, Vincenzo Roselli, Ignaz Rutter, Claudio Squarcella, Ioannis Tollis, and Sergio Tortora.

I would like to thank all the other people who have been part of our research group during these years, each of them contributed in creating the friendly and productive atmosphere that makes working in our office so great: Massimo Candela, Marco Chiesa, Marco Di Bartolomeo, Valentino Di Donato, Roberto di Lallo, Gabriele Lospoto,

Maurizio Pizzonia, Massimo Rimondini, Vincenzo Roselli, Claudio Squarcella, and Stefano Vissicchio.

I would like to thank Walter Didimo and Giuseppe Liotta for inviting me to the Bertinoro Workshop on Graph Drawing in 2013, 2014, and 2015, and Jan Kratochvìl for inviting me to the 2013 HOMONOLO Workshop. They gave the opportunity to work on new challenging problems with amazing people.

I would like to thank all my friends (both Series A and B, though this distinction faded away long ago). They always believed in me and never failed to encourage me to give my best.

My final heartfelt thanks go to those I love the most: My parents Angelo and Luisa for their unwavering trust and constant encouragement, my brother Paolo for the fundamental role he had in making me the person I am today, my nephews and niece for reminding me of the magic that lies inside every little gesture, my grandmother Anna for teaching me that kindness is all we need to win people's heart, and my soulmate Novella for making me feel like everything is possible. They are my life and the best things in it.

Contents

Introduction

The volume and the complexity of structured relational information created and handled by modern information systems has drastically increased. More and more often, data are generated so quickly that they cannon even be completely displayed or stored, and the manual analysis of their relational content may just be impracticable. In such a complex scenario, the creation of new paradigms and the design of efficient algorithms for the layout, the exploration and the analysis of large and dynamic graphs have emerged as strategic research directions, encompassed with both great practical interest and intrinsic theoretical appeal.

The classic notion of *graph*, as a mathematical abstraction used to describe a binary relation among a set of entities, has been extended to meet these new challenges. New *enhanced models* of graphs have been conceived. They allow for representing *aggregations*, *semantic affinities*, *hierarchies*, and *multiple* or even *dynamic relations* on the same set of entities.

It is not surprising, from a cognitive perspective, that the most natural and likely the most effective way to benefit from these combinatorial structures is to visualize them. In fact, drawings of graphs allow for quickly grasp their relational content. However, not all drawings might be equally effective to convey a graph's structure. *Graph Drawing* is the research field dealing with the design of algorithms to automatically produce visual representations of graphs. Among the features a "good" drawing of a graph should have, *planarity* is undoubtedly the most desired. In fact, drawings of a graph with no crossings among the edges are more readable and easier to navigate by the human eye. Hence, in order to produce nice and effective visualization of new models of graphs, the most promising direction is that of extending the standard notion of planarity also to such new models.

In particular, two new models of graphs that held the scene in the last years are the *clustered graphs* and the *simultaneous graphs*. A *clustered graph* consists of a graph and of a recursive hierarchical clustering of its vertex set. Clusters can be used to represent similarities among vertices, to abstract complex substructure into simpler

ones, and more in general to enrich a graph with hierarchical information. A *simultaneous graph* can be seen as a collection of graphs sharing the same vertex set or, alternatively, as a single graph with multiple edge sets. Simultaneous graphs allow to encode different relations among the same set of objects or the evolution of a single relation over time. For these models, visualization paradigms have been devised that are based on constraining the vertex set of the graph to lie in *prescribed regions* of the plane. For example, clusters are usually represented by enclosing vertices in simple closed regions, while the consistency between drawings of the constituent graphs of a simultaneous graph is achieved by mapping their vertices to the same set of points.

In this thesis, we deal with both clustered graphs and simultaneous graphs, and in particular with the two most famous notions of planarity for these models. Namely, CLUSTERED PLANARITY (C-PLANARITY for short) of clustered graphs and SIMULTANEOUS EMBEDDING WITH FIXED EDGES (SEFE for short) of simultaneous graphs. Further, we extend our focus, by considering as the possible embedding space for the vertices not only *simple nested regions* of the plane and *finite points sets*, but also *single lines*, *sets of parallel lines*, and *disjoint unbounded strips* of the plane. For example, in the framework of simultaneous graphs, we study problems PARTITIONED k-PAGE BOOK EMBEDDING, where vertices are forced to lie on a single line, T-COHERENT LEVEL PLANARITY, where vertices are forced to lie on parallel lines, and STREAMED PLANARITY where vertices are confined on a finite point set; further, in the framework of clustered graphs, we study STRIP PLANARITY, where vertices are assigned to disjoint unbounded regions of the plane; moreover, by requiring vertices to lie along parallel lines and, at the same time, inside nested regions intersecting those lines, we study CLUSTERED-LEVEL PLANARITY.

Clustered graphs were introduced together with the notion of C-PLANARITY, in which a clustered graph has to be drawn in the plane so that its drawing is planar and clusters are represented as simple regions enclosing all and only the vertices of the cluster in such a way that – to simplify – no "unnecessary" intersections involving clusters and edges are created. This notion of planarity aims at clearly displaying both the relational and the hierarchal information of the clustered graph at the same time. For simultaneous graphs, on the other hand, different notions of planarity were introduced. However, due to its practical and theoretical applications, SEFE is the most prominent one. In a SEFE a simultaneous graph has to be drawn in the plane in such a way that its drawing induces planar drawings of each of its constituent graphs and multi-edges are drawn as single edges. Alternatively, when considering each of such constituent graphs separately, a SEFE can be defined as a collection of planar drawings, one for each of such graphs, on the same set of points in which the common edges are represented by the same curves. This variant of planarity aims at clearly displaying each of the relations composing a simultaneous graph and, at the same

time, at helping the user maintain his/her mental map while exploring the structure of the graph.

Despite the great effort of research spent by the Graph Drawing community, determining the computational complexity of these problems is still among the most challenging and, even more so, fascinating open questions in this research field.

This thesis contributes to deepen the comprehension of the relationship between different and fundamental variants of planarity as well as to expand the knowledge about the computational complexity of some of these variants. We mainly deal with planar graphs (Part I), clustered graphs (Part II), strip graphs and clustered-level graphs (Part III), simultaneous graphs (Part IV), and streamed graphs (Part V). We have settled the question regarding the computational complexity of notable variants of planarity on which research had already been initiated in the last two decades. Further, we enriched the stage of planarity with new actors, by introducing and studying new constrained notions of planarity. Furthermore, we brought to light important combinatorial aspects and developed efficient algorithms for testing some of these old and new planarity problems.

Figure 0.1 provides an overview of some of the results presented in this thesis, where the emphasis is on the reducibility among different notions of planarity. Many algorithmic contributions are, however, not captured by this schema which is only meant as an updated map to help the reader explore the realm of the many declinations of planarity.

The thesis is organized as follows.

Part I deals with planar graphs, with the most common methods to described and handle their planar embeddings, and with the notions of planarity that are the main subject of this thesis.

In Chapter 1 we introduce some preliminaries and definitions about planar graphs, their embeddings, and their drawings.

In Chapter 2 we introduce basic concepts about the connectivity of graphs and we illustrate the main techniques for describing and handling the embeddings of planar graphs, depending on their degree of connectivity. In particular, we consider the *SPQR-trees* data structure that allows to efficiently handle the decomposition of a biconnected graph into its triconnected components.

In Chapter 3 we formally introduce the C-PLANARITY and SEFE problems and review the state of the art about the known polynomial-time algorithms to decide these problems for restricted classes of instances.

Part II deals with drawings of clustered graphs and constrained embeddings of planar multi-graphs.

In Chapter 4, in order to shed light on the C-PLANARITY problem we consider a relaxed version of it, where some kinds of crossings (either edge-edge, edge-region, or region-region) are allowed, even if the underlying graph is planar. In this setting, our main result is a polynomial-time algorithm to test whether a drawing with only region-region crossings exists for biconnected planar graphs, hence identifying a first non-trivial necessary condition for C-PLANARITY that can be tested in polynomial time for a noticeable class of clustered graphs.

In Chapter 5, motivated by the fact that the many computationally challenging problems as well as some of unknown complexity, like the C-PLANARITY problem, are polynomial-time solvable for embedded planar graphs with small faces, we study the problems MINMAXFACE and UNIFORMFACES of embedding a given biconnected multi-graph such that the largest face is as small as possible and such that all faces have the same size, respectively. Although both problems turn out to be computationally tough, a constant-factor approximation for MINMAXFACE and efficient algorithms for several interesting classes of instances of both problems are presented.

Part III deals with drawings of clustered graphs whose vertices are prescribed to lie in disjoint horizontal strips of the plane or on parallel lines.

In Chapter 6, we introduce and study a new notion of planarity, called STRIP PLANARITY, for graphs whose vertices are prescribed to lie within disjoint horizontal strips of the plane, called *strip graphs*. The problem has strong relationships with some of the most deeply studied variants of the planarity testing problem, such as C-PLANARITY, UPWARD PLANARITY, and LEVEL PLANARITY. Indeed, we show a polynomial-time reduction from this problem to the C-PLANARITY problem, thus relating their computational complexity. Further, we present a polynomial-time testing algorithm for embedded strip graphs, based on several augmentation techniques that eventually allow to reduce such instances to special instances of the UPWARD PLANARITY problem.

In Chapter 7, we study the computational complexity of two notorious notions of planarity related to the drawing of level graphs, that is, T-LEVEL PLANARITY and CLUSTERED-LEVEL PLANARITY. A level graph is a graph whose vertices are placed on parallel levels (or lines) with edges connecting only vertices on different levels. The former problem considers level graphs equipped, in the very same way as a clustered graph, with a recursive clustering of their vertex set. The latter problem, instead, considers level graphs in which each level is associated with a tree that imposes constrains on the ordering of the vertices on the corresponding level. Unlike the standard notion of LEVEL PLANARITY, we show that the additionally introduced constraints make these planarity problems NP-complete, while they become polynomial-time solvable when restricted to level graphs whose edges span only consecutive levels. It is worth

to point out that the algorithmic contribution is obtained by exploiting a non-trivial reduction between the two problems and their connection with the SEFE problem. We remark that, the NP-hardness of CLUSTERED-LEVEL PLANARITY presented in this chapter is, to the best of our knowledge, the first hardness result for a variation of the C-PLANARITY problem in which none of the c-planarity constraints is dropped.

Part IV deals with the SEFE problem, focusing in particular on the case in which the embedding space of the vertex set of the simultaneous graph is a single line, and with the relationship between the SEFE and C-PLANARITY problems.

In Chapter 8, we study the complexity of some combinatorial problems related to SEFE. Namely, we consider the variant of the problem called SUNFLOWER SEFE and PARTITIONED T-COHERENT k-PAGE BOOK EMBEDDING (PTBE-k for short) problems, which are known to be equivalent under certain conditions. We prove several hardness results for these problems even when simultaneous graphs composed of only thee planar graphs are considered and when strong restrictions on their degree of connectivity are imposed. In particular, our results on the PTBE-k problem for a fixed number of input graphs all sharing the same star subgraph, which is cited as a open problem by Schaefer [Sch13] and Hoske [Hos12], complements the work by Hong and Nagamochi [HN14] on this problem. On the positive side, we provide a linear-time algorithm for PTBE-k for an interesting class of simultaneous graphs composed of an arbitrary number of input graphs. Then, we focus on the still open case of SEFE where simultaneous graphs composed of only two input graphs are considered, called SEFE-2. First, we show that in the "connected" version of this problem, denoted C-SEFE-2, we can focus only on instances in which one of the two graphs has a special structure and high connectivity. Second, we study the optimization version of this problem where we seek to maximize the number of common edges that are drawn the same, called MAX SEFE, which is cited as an open question by Haeupler *et al.* [HJL13] and in Chapter 11 (Open Problem 9) of the Handbook of Graph Drawing and Visualization [BKR13b], and prove NP-completeness results for this problem in several restricted settings.

In Chapter 9 we deepen the understanding of the connection between the two long-standing Graph Drawing open problems that are the main subject of this thesis, that is, SEFE and C-PLANARITY. It is an intriguing question whether the two problems are polynomial-time equivalent. A major milestone in this respect was first proved by Schaefer [Sch13], who gave a reduction from C-PLANARITY to SEFE-2. In our research we make a further step to better comprehend the relationship between these problems, by showing that a reduction exists also in the opposite direction, if we consider instances of SEFE-2 in which the graph induced by the common edges is connected (C-SEFE-2).

Part V deals with drawings of non-planar graphs with crossing-free subgraphs and drawings of graphs that are presented in a streaming fashion.

In Chapter 10, we initiate the study of a new problem related to the SEFE and, particularly, to the SUNFLOWER SEFE problem. The focus of this research is on producing drawings of non-planar graphs in which a planar spanning subgraph is represented without intersections. In many application domains, edges might express connections between entities with different semantics and importance. In this case a visual interface might attempt to display more important edges without intersections, while disregarding intersections involving only the less important edges. Further, drawings in which edges are represented as straight-line segments, instead of polylines, are generally considered more readable. Hence, it also makes sense to consider drawings in which less important edges have bends, while still requiring the important edges to be represented as straight-line segments. Algorithms are presented for computing this kind of drawings for several classes of instances, both in the straight-line and in the polyline setting. In the former setting, we give positive and negative results for different kinds of connected spanning subgraphs and give an efficient testing algorithm for instances in which the crossing-free spanning subgraph is triconnected. In the latter setting, we provide different trade-offs between the number of bends and the required area of the drawing, and give a testing algorithm for instances in which the crossing-free spanning subgraph is biconnected and edges are allowed to have at most one bend.

In Chapter 11, we introduce and study a new model of graphs, called *streamed graphs*, that are graphs whose edges are presented as a stream evolving over time. We consider streamed graphs composed of edges that appear at a specific time instant and are visible for a limited time window before disappearing. Given a window size ω, we define the notion of ω-STREAM PLANARITY in which drawings for the edges of the streamed graph are searched for that induce crossing free drawing of the streamed graph at each time instant. It is not hard to see that this notion can be interpreted as a variant of SEFE. We show NP-completeness for this problem even when a constant window size is considered. Further, we consider the variant of the problem in which a given subgraph of the streamed graph, called *backbone*, never disappears, and show NP-completeness even for $\omega = 2$. On the positive side, we present a linear-time algorithm for this latter problem when $\omega = 1$, for which a polynomial-time algorithm was already given in the framework of PARTIAL PLANARITY [Sch14], whose study was motivated by the research presented in Chapter 10. Our results also imply that, unless P=NP, no FPT algorithms exist for the SEFE and the fundamental WEAK REALIZABILITY problem, with respect to some interesting parameters.

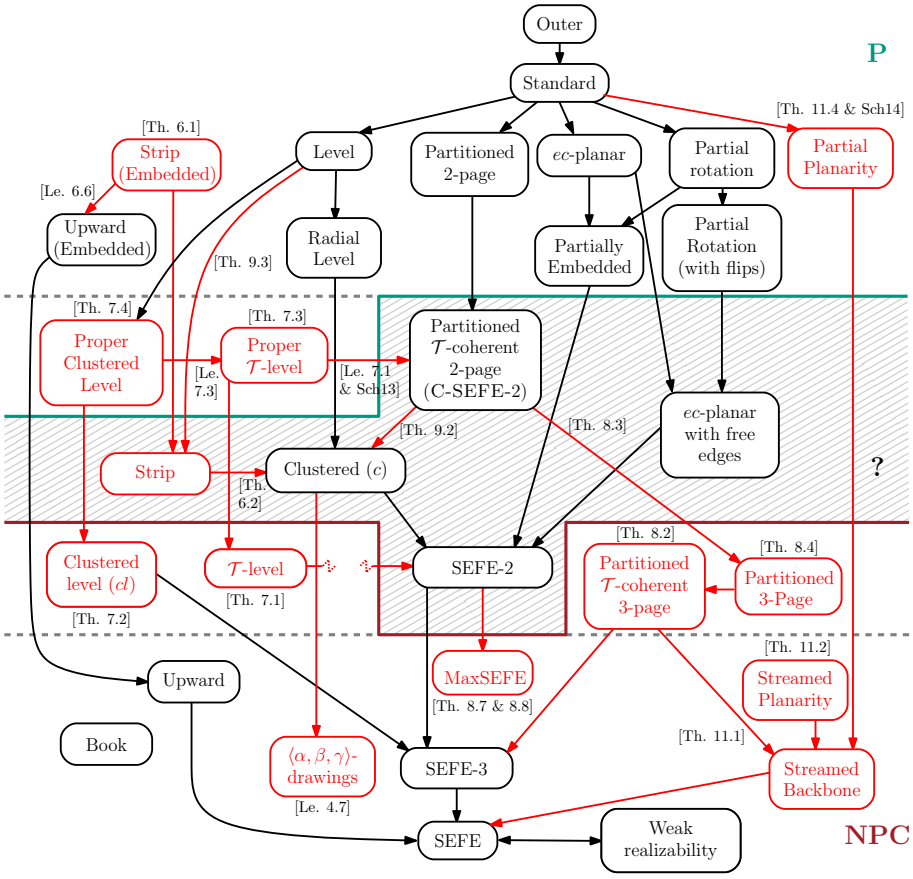


Figure 0.1: Updates on the classification proposed by Schaefer in [Sch13], where changes to the original schema in [Sch13] are emphasized by using the red color. Dashed lines represent the boundaries between problems that were known to be polynomial-time solvable, problems that were known to be NP-complete, and problems whose complexity was unknown before the work described in this thesis. Boxes for newly introduced problems have been added. Solid lines show the new boundaries according to the results of this thesis. The arcs representing reductions that can be transitively inferred are omitted. Arcs connecting a more constrained version of a problem to its generalization are also drawn. Results described in this thesis are equipped with references.

Part I

Preliminaries & Background

Chapter 1

Graph Preliminaries and Definitions

In this chapter, we introduce some preliminaries and definitions about graphs, their embeddings, and their drawings that are used throughout this thesis.

A reader who wants to assume more familiarity with the basic concepts about graphs, algorithms, and geometry, may refer to books on Graph Theory (e.g., [Har69, BM76, CN88, Die05]), to books on Algorithms (e.g., [Eve79, GJ83, AHU83, CLRS09, GT09]), and to books on Computational Geometry (e.g., [PS85, Ede87, dCvO08]). Reference books containing detailed definitions, basic concepts, and most important results about Graph Drawing can also be useful and interesting to read [KW01, NR04, DETT99].

The chapter is structured as follows. In Section 1.1 we give basic definitions about graphs. Then, in Section 1.2 we deal with planar graphs and planar embeddings, and define some interesting subclasses of planar graphs. Finally, in Section 1.3 we describe the main drawing conventions and aesthetic criteria used in Graph Drawing.

1.1 Basic Definitions

A *graph* $G = (V, E)$ is composed of a set V of *vertices* or *nodes*, and a multiset E of *unordered* pairs of vertices, called *edges* or *arcs*. Given an edge $e = (u, v) \in E$, we say that u and v are *incident* to e (u and v are the *end-vertices* of e), and that e is *incident* to u and v. Also, we say that two vertices are *adjacent* if they are incident to the same edge, and two edges are *adjacent* if they are incident to the same vertex. The *degree of a vertex* v, denoted by $\deg(v)$, is the number of edges incident to it. The *degree of a graph* G, denoted by $\Delta(G)$, is the maximum among the degrees of its vertices.

A graph $G = (V, E)$ is *directed*, also referred to as *digraph*, if the pairs of vertices in E are ordered. In a directed graph, an edge (v, w) is *oriented* from its *tail* (or *origin*) v to its *head* (or *end*) w; also, the edge is *outgoing* from v and *incoming* to w. The *indegree* of a vertex v is the number of its incoming incident edges; analogously, the *outdegree* is the number of its outgoing edges. A vertex whose indegree equals 0 is called *source*; analogously, a vertex whose outdegree equals 0 is called *sink*. The graph obtained from a digraph G by considering its edges without orientation is called the *underlying graph* of G.

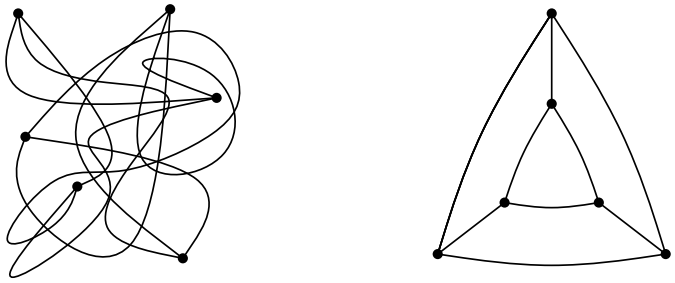
A *self-loop* in a graph $G = (V, E)$ is an edge $(u, u) \in E$. A set of *multiple edges* in a graph (V, E) is a set of edges connecting the same two vertices $u, v \in V$. A graph is *simple* if it contains neither self-loops nor multiple edges. In the following, unless otherwise specified, we always refer to simple undirected graphs.

A graph is k*-regular* if all its vertices have degree k, that is, it holds that $\deg(v) = k$, for each vertex $v \in V$. A 3-regular graph is also called *cubic* and a graph G is *subcubic* if all its vertices have degree at most 3, that is, $\Delta(G) \leq 3$.

A graph $G' = (V', E')$ is a *subgraph* of a graph $G = (V, E)$ if $V' \subseteq V$ and $E' \subseteq E$. We say that $G' = (V', E')$ is *induced* by V' if, for each edge $(u, v) \in E$ such that $u, v \in V'$, we have $(u, v) \in E'$. Also, $G' = (V', E')$ is a *spanning subgraph* of $G = (V, E)$ if it is a subgraph of G and $V' = V$. A graph $G' = (V', E')$ is a *supergraph* of a graph $G = (V, E)$ if $V \subseteq V'$ and $E \subseteq E'$. A graph H is a *proper* subgraph (supergraph) of a graph G if G contains at least a vertex or an edge more (less) than H. A subgraph H of G is *maximal* under some condition $\rfloor$ if there exists no subgraph H' of G satisfying condition $\rfloor$ such that H' is a proper supergraph of H.

A graph $G = (V, E)$ is *complete* if for each pair of vertices $u, v \in V$, edge $(u, v) \in E$. The complete graph on n vertices is denoted by K_n. A graph is *bipartite* if V can be partitioned into two sets V_1 and V_2 such that for each edge $(u, v) \in E$, either $u \in V_1$ and $v \in V_2$ or vice versa. A bipartite graph is complete if for each $v_i \in V_1$ and for each $v_j \in V_2$, edge $(v_i, v_j) \in E$. Complete bipartite graphs are denoted by $K_{a,b}$, where $a = |V_1|$ and $b = |V_2|$.

A *subdivision* of a graph G is a graph G' that can be obtained by replacing each edge of G with a path of arbitrary length. The *contraction* of an edge (u, v) consists of the replacement of u, v, and (u, v) with a single vertex w, of each edge (u, z) with an edge (w, z), and of each edge (v, z') with an edge (w, z'). A *minor* of a graph G is any graph that can be obtained from G by a sequence of removals of vertices, removals of edges, and contractions of edges.



(a) (b)

Figure 1.1: (a) A non-planar drawing of a graph. (b) A planar drawing of the same graph.

1.2 Planar Graphs

Planar graphs are probably the most studied class of graphs in Graph Theory, and surely the most studied class of graphs in Graph Drawing. In this section we give preliminaries and definitions about planar graphs.

A *drawing* of a graph is a mapping of each vertex to a distinct point of the plane and of each edge to a simple Jordan curve connecting the points to which the end-vertices of the edge have been mapped. A drawing is *planar* if the curves representing edges do not cross except, possibly, at common endpoints. A graph is *planar* if it admits a planar drawing. A planar drawing of a graph provides extremely high readability of the combinatorial structure of the graph [PCJ97, Pur00]. See 1.1 for a comparison between a non-planar and a planar drawing.

A planar drawing of a graph determines a clockwise circular ordering of the edges incident to each vertex, called *rotation scheme* of the vertex. Two drawings of the same graph are *equivalent* if they determine the same rotation scheme for each vertex. A *combinatorial embedding* (or simply *planar embedding* or *embedding*) is an equivalence class of planar drawings. A graph is *embedded* when an embedding of it has been decided. A planar drawing partitions the plane into topologically connected regions, called *faces*. A vertex (an edge) is *incident* to a face f if it belongs to sequence of vertices (edges) delimiting f. All the faces are bounded, except for one face, that we call the *outer face* (or *external face*). The other bounded faces are called *internal faces*. Two planar drawings with the same combinatorial embedding have the same faces. However, such drawings could still differ for their outer faces. Refer to Fig. 1.2.

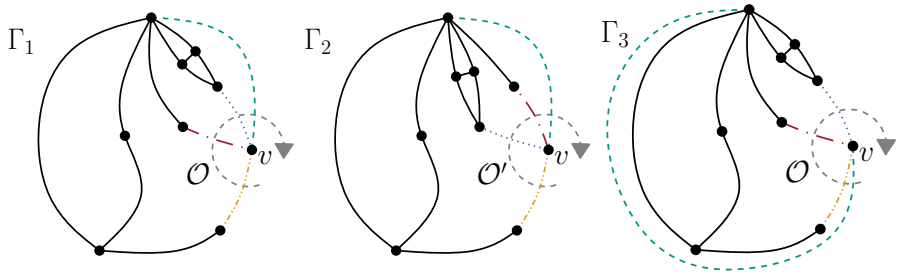


Figure 1.2: Three drawings Γ_1, Γ_2, and Γ_3 of the same planar graph, focused on the rotation scheme of vertex v. Drawings Γ_1 and Γ_2 have different combinatorial embeddings; drawings Γ_1 and Γ_3 have the same combinatorial embedding but a different external face. The rotation schemes of vertex v in Γ_1 (Γ_3) and in Γ_2 are denoted by $\mathcal{O}$ and $\mathcal{O}'$, respectively.

A graph together with a planar embedding and a choice for its outer face is called a *plane graph*. In a plane graph, *external* and *internal* vertices are defined as the vertices incident and not incident to the outer face, respectively. Analogously, *external* and *internal* edges are defined as the edges incident and not incident to the outer face, respectively. Sometimes, the distinction is made between *planar embedding* and *plane embedding*, where the former is an equivalence class of planar drawings and the latter is a planar embedding together with a choice for the outer face. When such a distinction is made, then a planar embedding is more commonly referred to as a combinatorial embedding.

The *dual graph* $G^\star$ of an embedded planar graph G is a planar multi-graph having a vertex v_f for each face f of G and an edge (v_f, v_g) if and only if faces f and g of G share an edge. We say that edge (v_f, v_g) is the *dual edge* of e, and vice versa. Fig. 1.3 shows an embedded planar graphs and its dual graph. It is important to notice that the dual graph of an embedded planar graph depends on the embedding of the graph, that is, different dual graphs correspond to different planar embeddings of the same graph. On the other hand, the dual graph of an embedded planar graph G is independent of the choice of the outer face of G.

A plane graph is *maximal* (or equivalently is a *triangulation*) if all its faces are delimited by cycles of three vertices. A planar graph is *maximal* if it admits a planar embedding that determines a triangulation. A triangulation is maximal in the sense that adding an edge to it yields a non-planar graph. Maximal planar graphs are an important and deeply studied class of planar graphs since any planar graph can be

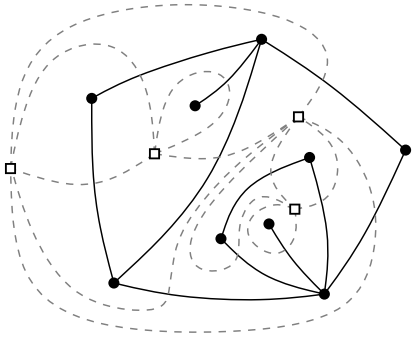


Figure 1.3: A plane graph whose vertices are drawn as black disks and whose edges are drawn as solid curves, and its dual graph, whose vertices are drawn as white boxes and whose edges are drawn as dashed curves.

.

augmented to maximal by adding *dummy edges* to it and since triangulations, as the (subdivisions of) triconnected planar graphs, admit exactly one combinatorial embedding (a more detailed discussion about the properties of highly-connected planar graphs is given in Chapter 2) and hence are often easier to deal with. A plane graph is *internally-triangulated* when all its internal faces have exactly three incident vertices.

Planarity

Planarity is commonly accepted as the most important aesthetic criteria a drawing should satisfy to be nice and readable. In fact, the absence of partial or complete overlapping among the objects makes the drawing aesthetically pleasant and easily readable by the human eye, and provides extremely high readability of the combinatorial structure of the graph, as confirmed by some cognitive experiments in graph visualization [PCJ97, Pur00, PCA02, WPCM02]. However, the great importance of planar graphs, so in Graph Drawing as in Graph Theory and Computational Geometry in general, also comes from the many mathematical, combinatorial, and geometrical properties they exhibit.

From the combinatorial and topological point of view, the first important result about planar graphs is the characterization given by Kuratowski [Kur30] in 1930, stating that a graph is planar if and only if it contains no subgraph isomorphic to a subdivision of the complete graph K_5 with five vertices or to a subdivision of the complete bipartite graph $K_{3,3}$ with three vertices in each of the sets of the bipartition.

Afterwards, such a characterization has been extended by Wagner, who stated that a graph is planar if and only if it contains no K_5-minor and no $K_{3,3}$-minor [Wag37].

The planarity of a graph can be tested in linear time, as first shown by Hopcroft and Tarjan [HT74] in 1974. Linear-time algorithms for testing the planarity of a graph have been also presented, e.g., in [BL76, ET76, dR82, BM04, dFOdM12, HT08]. Such testing algorithms can be suitably modified in order to compute planar embeddings if the test yields a positive result. If an embedding of a graph is fixed, then linear time still suffices to test if the embedding is planar [Kir88].

The following result, known as *Euler's formula*, relates the number of vertices, edges, and faces in a connected plane graph.

Theorem 1.1 (Euler 1750) *Let G be a connected plane graph, and let n, m, and f denote the number of vertices, edges, and faces of G, respectively. Then, the following equation holds:*

$$n - m + f = 2$$

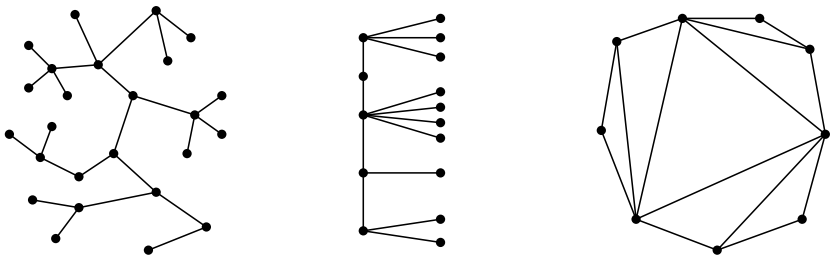
The fact that the planarity testing, so as many other problems on planar graphs, can be solved in linear time is due to an important mathematical property of planar graphs, stating that the number of edges of a planar graph is linear in the number of its vertices. Namely, by the Euler's formula, we have $m \leq 3n - 6$, where m is the number of edges, in any n-vertex planar graph.

Classes of Planar Graphs

In this section we present preliminaries and definitions about some important subclasses of planar graphs that will be used in the rest of the thesis.

A *cycle* is a connected graph such that each vertex has degree exactly two. A *tree* is a connected acyclic (i.e., not containing any cycle) graph (see Fig. 1.4(a)). A *pseudo-tree* is a connected graph containing exactly one cycle. A *path* is a tree such that each vertex has degree at most two. A *chord* of a cycle (of a path) is an edge connecting two non-consecutive vertices of the cycle (of the path). The degree-1 vertices of a (pseudo-)tree are its *leaves*, while the other vertices are its *internal* vertices. We denote the set of leaves of a (pseudo-)tree T by $\mathcal{L}(T)$. A *leaf edge* is an edge incident to a leaf. A *caterpillar* (see Fig. 1.4(b)) is a tree such that removing all the leaves and all the leaf edges yields a path, called *spine* of the caterpillar, whose nodes and edges are called *spine nodes* and *spine edges*, respectively. A *star graph* is a tree such that removing all leaves and all leaf edges yields an isolated node, called *central node*.

A *rooted tree* is a tree with one distinguished node, called *root*. In a rooted tree, the *depth* of a node v is the length of the unique path (i.e., the number of edges composing



(a) (b) (c)

Figure 1.4: (a) A tree. (b) A caterpillar. (c) A maximal outerplane graph.

the path) between v and the root. The *depth* of a rooted tree is the maximum depth among all the vertices. A *binary tree* (a *ternary tree*) is a rooted tree such that each node has at most two children (resp. three children). A tree is *ordered* if an order of the children of each node (i.e., a planar embedding) is specified. In an ordered binary tree we distinguish the *left* and the *right child* of a node. The *subtrees* of a node u of a tree T are the subtrees of T rooted at u and not containing the root of T.

An *outerplane graph* is a plane graph such that all the vertices are incident to the outer face. An *outerplanar embedding* is a planar embedding such that all the vertices are incident to the same face. An *outerplanar graph* is a graph that admits an outerplanar embedding. From a combinatorial point of view, an outerplanar graph is a graph that contains no K_4-minor and no $K_{2,3}$-minor. Also, outerplanar graphs have at most $2n - 3$ edges. Note that trees and cycles are outerplanar graphs. A *maximal outerplane graph* (see Fig. 1.4(c)) is an outerplane graph such that all its internal faces are delimited by cycles of three vertices. A *maximal outerplanar embedding* is an outerplanar embedding such that all its faces, except for the one to which all the vertices are incident, are delimited by cycles of three vertices. A *maximal outerplanar graph* is a graph that admits a maximal outerplanar embedding. Every outerplanar graph can be augmented to maximal by adding dummy edges to it.

A *series-parallel graph (SP-graph)* is a graph with no K_4-minor. SP-graphs are inductively defined as follows. An edge (u, v) is a series-parallel graph with *poles* u and v. Denote by u_i and v_i the poles of a series-parallel graph G_i. A *series composition* of SP-graphs $G_1, \ldots, G_k$, with $k \geq 2$, is an SP-graph with poles $u = u_1$ and $v = v_k$, containing graphs G_i as subgraphs, and such that v_i and u_{i+1} have been identified, for each $i = 1, \ldots, k - 1$ (see Fig. 1.5(a)). A *parallel composition* of SP-graphs $G_1, \ldots, G_k$, with $k \geq 2$, is an SP-graph with poles $u = u_1 = \cdots = u_k$

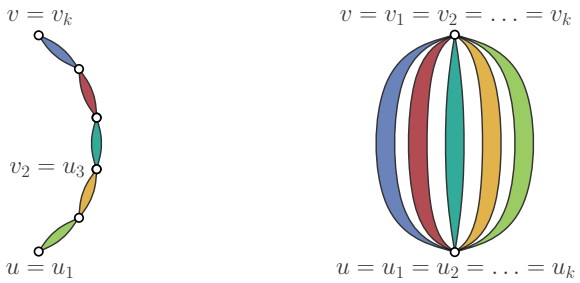


Figure 1.5: (a) A series composition of a sequence $G_1, G_2, \ldots, G_k$ of series-parallel graphs. (b) A parallel composition of a set $G_1, G_2, \ldots, G_k$ of series-parallel graphs.

and $v = v_1 = \cdots = v_k$ and containing graphs G_i as subgraphs (see Fig. 1.5(b)). If follows that any (connected) subgraph of a series-parallel graph is a series-parallel graph. Observe that outerplanar graphs are series-parallel graphs.

A graph G is a *k-tree* if it can be generated by a sequential addition of vertices (and their incident edges) in an order $v_1, \ldots, v_n$ such that, for each $i > k$, vertex v_i has exactly k *predecessors* (i.e, neighbors with smaller index) and they form a *clique* (i.e, the subgraph of G induced by the predecessors of v_i is the complete graph K_k). A *partial k-tree* is a subgraph of a k-tree and have *treewidth* (i.e., the size of the largest clique in the graph) bounded by a constant. Partial k-trees received large attention since, as their treewidth is bounded by a constant, they allow for solving in polynomial time problems that are otherwise NP-hard ([AP89, CR05]). Trees coincide with 1-trees, while maximal series-parallel graphs coincide with 2-trees. *Planar 3-trees*, also referred as *stacked triangulations* or *Apollonian graphs*, are special types of planar triangulations which can be generated from a triangle by a sequential addition of vertices of degree 3 inside faces. Namely, planar 3-trees can be inductively defined as follows:

- The complete graph K_3 on three vertices is a planar 3-tree.
- Let G be a planar 3-tree with n vertices and let a, b, and c be three vertices bounding a face of G. The graph G' obtained by adding vertex v and edges (a, v), (b, v), and (c, v) is a planar 3-tree with $n + 1$ vertices.

A graph G with $n \geq 4$ vertices is a *wheel* W_n if it consists of a simple cycle $C = v_1, \ldots, v_{n-1}$ on $(n - 1)$ vertices and of a vertex, called the *central vertex*, that

is connected to all the vertices of C. Wheels are triconnected graphs. The complete graph on four vertices K_4 is a wheel W_4 on four vertices.

1.3 Graph Drawing

In this section, we introduce basic concepts about Graph Drawing. A Graph Drawing algorithm takes as an input a graph G and outputs a *nice* drawing of G. What makes a drawing "nice", is the fact that it is easily understandable by the human eyes, that is, the fact that it is *readable*. In order to construct a nice drawing of the graph, it is important the knowledge of the class of graphs G belongs to. In fact, several graph drawing algorithms only work for restricted classes of graphs. Moreover, the drawing should reflect the combinatorial properties of the graph, which are mainly encoded in the class of graphs G belongs to. It is also important to observe that the best drawing of a graph might not exist. In fact, different individuals usually have different perceptions of the same drawing; moreover, different domains of applications determine different requirements for the drawings. The requirements a drawing must satisfy in order to be admissible are generally regarded as the *drawing conventions*; the properties that a drawing should satisfy as much as possible are generally regarded as the *drawing aesthetics*.

In the reminder of the section, we describe the most used drawing conventions and discuss some aesthetic criteria that characterize a good drawing of a graph.

Drawing Conventions

When aiming at high readability of a drawing, an important issue that has to be considered concerns the geometrical representation of the edges and of the faces. Namely, planar drawings in which edges are represented by straight-line segments (known as *straight-line drawings*, see Fig. 1.6(a)) happen to be more readable than drawings in which edges are represented by poly-lines (known as *poly-line drawings*, see Fig. 1.6(b)) or general curves, and drawings in which faces are drawn as convex polygons (known as *convex drawings*, see Fig. 1.6(c)) are more readable than drawings in which this is not the case (see Fig. 1.6(a)). Among the more used and studied drawing conventions, we also mention *orthogonal drawings*, in which each edge is represented by a sequence of horizontal and vertical segments.

Other drawing conventions that are worth to mention are the *grid drawings*, in which vertices and bends have integer coordinates, *upward drawings* of digraphs, in which each edge is represented by a curve monotonically-increasing in the upward direction, and *proximity drawings*, in which given a definition of *proximity*, the *proximity graph* of a set of points is the graph with a vertex for each point of the set, and

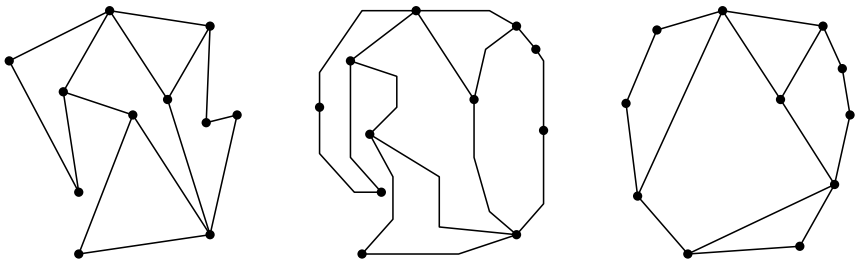


Figure 1.6: (a) A straight-line planar drawing of a planar graph G. (b) A poly-line planar drawing of G. (c) A convex drawing of G.

with an edge between two vertices if the corresponding points satisfy the proximity property. Then, a proximity drawing of a graph G is a drawing D of G such that the proximity graph of the set of points on which the vertices of G are drawn in D coincides with G itself. An example of proximity graphs is the *Delaunay triangulation* for a set P of points in the plane, that is, a triangulation T such that no point in P is inside the circumscribed circle of any triangle in T.

The most studied and used drawing convention is the one of straight-line drawings. Of course such a convention is much more restrictive than the one in which edges can have bends, and hence many results that hold for poly-line drawings do not hold for straight-line drawings. However, regarding planarity, this is not the case. Indeed, a very important result, known as Fary's theorem and independently proved by Wagner [Wag36], by Fary [Fár48], and by Stein [Ste51], states that a graph admits a straight-line planar drawing if and only if it admits a planar drawing. This result shows that planarity does not depend on the geometry used for representing the edges but it only depends on the topological properties of the graph.

Aesthetic Criteria

Some aesthetic criteria can be defined to measure the quality of a drawing. They are naturally associated with optimization problems most of which are computationally hard to solve. Among such criteria, one of the most important is certainly the area occupied by the drawing, that is, the area of the smallest rectangle with sides parallel to the coordinate axes that contains all the drawing. Of course, small area drawings can not be obtained by simply scaling down the drawing, since some *resolution rules* have

to be respected in the drawing for maintaining readability. In particular, a minimum distance, say one unit, between two elements (vertices and edges) of the drawing has to be maintained. In order to respect some of such rules, when dealing with area minimization problems, vertices are usually placed on an integer grid, in such a way that the minimum distance between any two of them is at least one grid unit. In this direction, it has been shown in several papers that every n-vertex plane graph admits a planar straight-line drawing on a $O(n^2)$ area grid [dPP88, dPP90, Sch90, CN98, ZH03, BFM07]. Further, a grid of quadratic size is asymptotically the best possible for straight-line planar drawings, since there exist planar graphs requiring such an area in any planar grid drawing [Val81, dPP90, FP08].

Further examples of commonly used aesthetic criteria are the following:

- *Aspect Ratio*: Minimization of the *aspect ratio* of the drawing. The aspect ratio of a drawing is the ratio between the longest and the smallest side of the bounding box of the drawing.
- *Angular Resolution*: Maximization of the smallest angle between two edges incident to the same vertex.
- *Crossings*: Minimization of the number of crossings in the drawing. Notice that this aesthetic does not make sense when the drawing is required to be planar.
- *Total Edge Length*: Minimization of the sum of the lengths of the edges in the drawing.
- *Total Bends*: Minimization of the number of bends in the drawing.

Chapter 2

Data Structures

In this chapter we describe useful data structures that will be exploited throughout this thesis. Data structures exist to describe and efficiently handle the decomposition of general graphs into their components of higher degree of connectivity for some values of connectivity. We present two such data structures for connected and biconnected graphs, respectively. Namely, we describe block-cutvertex trees (BC-trees) in Section 2.2 and SPQR-trees in Section 2.3. Further, in Section 2.4 we describe PQ-trees, a data structure that allows to succinctly represent a family of permutations of a set of objects A with consecutivity constraints among subsets of objects in A.

2.1 Connectivity

A graph $G = (V, E)$ is *connected* if, for each pair of vertices $u, v \in V$, there exists a path connecting u and v. A graph that is not connected is said to be *disconnected*. More generally, a graph $G = (V, E)$ is *k-connected* if, for each pair of vertices $u, v \in V$, there exist k vertex-disjoint paths connecting u and v. Alternatively, a graph $G = (V, E)$ is *k-connected* if it remains connected after the removal of any set of $k - 1$ vertices; 3-connected, 2-connected, and 1-connected graphs are also called *triconnected*, *biconnected*, and *simply connected* graphs, respectively. A *separating k-set* is a set of k vertices whose removal disconnects the graph. Separating 1-sets, separating 2-sets, and separating 3-sets are also called *cutvertices*, *separation pairs*, and *separation triples*, respectively. Hence, a connected graph is biconnected if it has no cutvertices, a biconnected graphs is triconnected if it has no separation pairs, and a triconnected graph id 4-connected if it has no separation triples. The maximal biconnected subgraphs of a graph are its *blocks*, while the maximal triconnected subgraphs

of a graph are its *triconnected components*. Each edge of G falls into a single block of G and into a single triconnected component, while cutvertices are shared by different blocks and vertices belonging to a separation pair are shared by different separation pairs. A *cut* of a graph $G = (V, E)$ is a partition if its vertices into two subsets V_1 and V_2. A *cutset* of G is the set $E' \subseteq E$ such that $(u, v) \in E'$ if and only if $u \in V_1$ and $v \in V_2$. Observe that the removal of the edges of E' from E increases the number of connected components of G.

Connectivity, Planarity, and Combinatorial Embeddings

In this subsection we discuss some properties of planar graphs and their combinatorial embeddings that depend on the degree of connectivity of the graph.

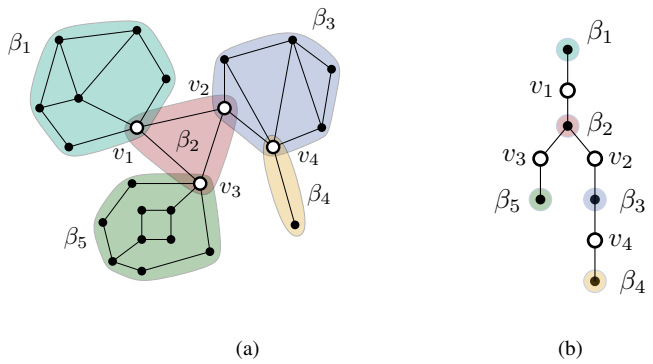
In general a connected planar graph G has several combinatorial embedding.

Let G be a simply-connected planar graph, that is, G has a cutvertex v, and let $H_1, \ldots, H_l$ be the connected components of graph $H = G \setminus v$. Also, let $\mathcal{E}$ be any combinatorial embedding of G. Then, it is possible to construct a new combinatorial embedding $\mathcal{E}'$ of G from $\mathcal{E}$ by replacing a component H_j of H (together with the edges connecting v to the vertices of H_j) lying in face f of $\mathcal{E}$ in another face $f' \neq f$ of $\mathcal{E}$ incident to v. Similarly, let G be a biconnected planar graph, that is, G has a separation pair $\{u, v\}$, and let $H_1, \ldots, H_l$ be the connected components of graph $H = G \setminus \{u, v\}$. Then, it is possible to construct a new combinatorial embeddings of G by flipping or by reordering some of $H_1, \ldots, H_l$ (together with their edges connecting u and v to the vertices of each component) around $\{u, v\}$.

A theorem by Whitney [Whi33] states that every 3-connected planar graph admits exactly two combinatorial planar embeddings, which are one the flip of the other, that is, the rotation schemes of the vertices in the two embeddings are reversed. Since any $k + 1$-connected graph is also k-connected, this result implies that no embedding choice (except for a flip of the embedding) exists for higher degree planar graphs. However, it is worth pointing out that there exists no k-connected planar graph with $k \geq 6$. This result can be easily derived from the fact a graph is k-connected only if its minimum degree is at least k and by the following simple property that can be proved by exploiting Euler's formula.

Property 2.1 *Every planar graph contains a vertex of degree at most* 5.

We mention another important relation between connectivity and planarity. Namely, a simply-connected (resp., biconnected) graph G is planar if and only if all its blocks (resp., triconnected components) are planar [Har69]([BDD00]).



(a) (b)

Figure 2.1: (a) A connected graph and (b) its BC-tree, rooted at block β_1.

Problems concerning the research of a particular embedding of a planar graph satisfying specific properties that are difficult for general planar graphs, can be often efficiently solved when the graph is triconnected. Hence, a possible strategy to tackle such problems in general low-connected planar graphs is to first decompose such graphs into their highly-connected components, as the problem can be solved more easily on such components, and then efficiently combine the solutions for such (smaller and highly-connected) components exploiting the property of the decomposition. The next sections will be devoted to two such decompositions for simply connected and biconnected graphs, respectively.

2.2 BC-trees

To handle the decomposition of a connected graph into its biconnected components, we use *block-cutvertex trees* (usually referred to as *BC-trees*), a data structure introduced by Harary and Prins in [HP66].

The *BC-tree* $\mathcal{T}$ of a connected graph $G(V, E)$ is a tree with a B-node for each block of G and a C-node for each cutvertex of G. Edges in $\mathcal{T}$ connect each B-node β to the C-nodes associated with the cutvertices in the block of β. We consider BC-trees that are rooted at a specific block of G.

The *size* of $\mathcal{T}$ is the number of nodes of $\mathcal{T}$, which is equal to the number of blocks plus the number of cutvertices. Hence, the size of $\mathcal{T}$ is $O(n)$, where $n = |V|$ is the number of vertices of G. Figure 2.1 shows a connected graph and its BC-tree, rooted at block β_1.

The first algorithm to decompose a connected undirected graph into its blocks is due to Hopcroft and Tarjan [HT73] and runs in linear time. Tarjan and Vishkin [TV85] also presented a parallel algorithm that runs in $O(\log n)$ time on a concurrent-read concurrent-write parallel RAM (CRCW PRAM) with $n + m$ processors, where n and m are the number of vertices and edges of the considered graph, respectively.

For some specific examples of applications of this data structure the interested reader might refer, for instance, to [PT00b, DDLM05, CDF$^+$08, AFG10]; see also Chapter 11.

2.3 SPQR-trees

To handle the decomposition of a biconnected graph into its triconnected components, we use *SPQR-trees*, a data structure introduced by Di Battista and Tamassia in [DT90, DT96a, DT96b].

A graph is *st-biconnectible* if adding edge (s, t) to it yields a biconnected graph. Let G be an st-biconnectible graph. A *separation pair* of G is a pair of vertices whose removal disconnects the graph. A *split pair* of G is either a separation pair or a pair of adjacent vertices. A *maximal split component* of G with respect to a split pair $\{u, v\}$ (or, simply, a maximal split component of $\{u, v\}$) is either an edge (u, v) or a maximal subgraph G' of G such that G' contains u and v, and $\{u, v\}$ is not a split pair of G'. A vertex $w \neq u, v$ belongs to exactly one maximal split component of $\{u, v\}$. We call *split component* of $\{u, v\}$ the union of any number of maximal split components of $\{u, v\}$.

We consider SPQR-trees that are rooted at one edge of the graph, called the *reference edge*. The rooted SPQR-tree $\mathcal{T}$ of a biconnected graph G, with respect to a reference edge e, describes a recursive decomposition of G induced by its split pairs. The nodes of $\mathcal{T}$ are of four types: S, P, Q, and R. Their connections are called *arcs*, in order to distinguish them from the edges of G.

Each node τ of $\mathcal{T}$ has an associated st-biconnectible multigraph, called the *skeleton* of τ and denoted by sk(τ). Skeleton sk(τ) shows how the children of τ, represented by "virtual edges", are arranged in τ. The virtual edge in sk(τ) associated with a child node σ, is called the *virtual edge of σ in sk(τ)*.

For each virtual edge e_i of sk(τ), recursively replace e_i with the skeleton sk(τ_i) of its corresponding child τ_i. The subgraph of G that is obtained in this way is the *pertinent graph* of τ and is denoted by pert(τ).

For each virtual edge e_i of sk(τ), the *expansion graph* $exp(e_i)$ of e_i is the pertinent graph $pert(\mu_i)$ of the child μ_i of τ represented by edge e_i in sk(τ).

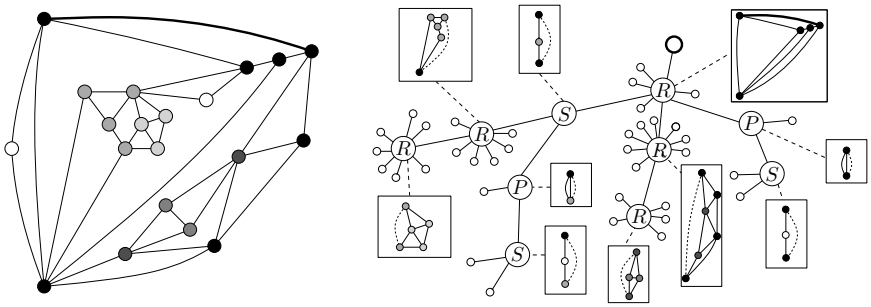
Given a biconnected graph G and a reference edge $e = (u', v')$, tree $\mathcal{T}$ is recursively defined as follows. At each step, a split component G^*, a pair of vertices $\{u, v\}$, and a node σ in $\mathcal{T}$ are given. A node τ corresponding to G^* is introduced in $\mathcal{T}$ and attached to its parent σ. Vertices u and v are the *poles* of τ and denoted by $u(\tau)$ and $v(\tau)$, respectively. The decomposition possibly recurs on some split components of G^*. At the beginning of the decomposition $G^* = G - \{e\}$, $\{u, v\} = \{u', v'\}$, and σ is a Q-node corresponding to e.

Base Case: If G^* consists of exactly one edge between u and v, then τ is a Q-node whose skeleton is G^* itself.

Parallel Case: If G^* is composed of at least two maximal split components $G_1, \ldots, G_k$ ($k \geq 2$) of G with respect to $\{u, v\}$, then τ is a P-node. Graph sk(τ) consists of k parallel virtual edges between u and v, denoted by $e_1, \ldots, e_k$ and corresponding to $G_1, \ldots, G_k$, respectively. The decomposition recurs on $G_1, \ldots, G_k$, with $\{u, v\}$ as pair of vertices for every graph, and with τ as parent node.

Series Case: If G^* is composed of exactly one maximal split component of G with respect to $\{u, v\}$ and if G^* has cutvertices $c_1, \ldots, c_{k-1}$ ($k \geq 2$), appearing in this order on a path from u to v, then τ is an S-node. Graph sk(τ) is the path $e_1, \ldots, e_k$, where virtual edge e_i connects c_{i-1} with c_i ($i = 2, \ldots, k-1$), e_1 connects u with c_1, and e_k connects c_{k-1} with v. The decomposition recurs on the split components corresponding to each of $e_1, e_2, \ldots, e_{k-1}, e_k$ with τ as parent node, and with $\{u, c_1\}, \{c_1, c_2\}, \ldots, \{c_{k-2}, c_{k-1}\}, \{c_{k-1}, v\}$ as pair of vertices, respectively.

Rigid Case: If none of the above cases applies, the purpose of the decomposition step is that of partitioning G^* into the minimum number of split components and recurring on each of them. We need some further definition. Given a maximal split component G' of a split pair $\{s, t\}$ of G^*, a vertex $w \in G'$ *properly belongs* to G' if $w \neq s, t$. Given a split pair $\{s, t\}$ of G^*, a maximal split component G' of $\{s, t\}$ is *internal* if neither u nor v (the poles of G^*) properly belongs to G', *external* otherwise. A *maximal split pair* $\{s, t\}$ of G^* is a split pair of G^* that is not contained into an internal maximal split component of any other split pair $\{s', t'\}$ of G^*. Let $\{u_1, v_1\}, \ldots, \{u_k, v_k\}$ be the maximal split pairs of G^* ($k \geq 1$) and, for $i = 1, \ldots, k$, let G_i be the union of all the internal maximal split components of $\{u_i, v_i\}$. Observe that each vertex of G^* either properly belongs to exactly one G_i or belongs to some maximal split pair $\{u_i, v_i\}$. Node τ is an R-node. Graph sk(τ) is the graph obtained from G^* by replacing each



(a) (b)

Figure 2.2: (a) A biconnected planar graph and (b) its SPQR-tree, rooted at a Q-node adjacent to the R-node whose internal vertices are black. The skeletons of the internal R-nodes of the tree are represented inside the boxes. The virtual edge representing the parent of a node μ in the skeleton of μ is drawn as a dotted line.

subgraph G_i with the virtual edge e_i between u_i and v_i. The decomposition recurs on each G_i with μ as parent node and with $\{u_i, v_i\}$ as pair of vertices.

For each node τ of $\mathcal{T}$, the construction of sk(τ) is completed by adding a virtual edge (u, v) representing the rest of the graph. Fig. 2.2 depicts a biconnected planar graph and its SPQR-tree.

In this thesis we will be mostly concerned with rooted SPQR-trees, however, it is worth pointing out that when using a different reference edge e' of G a decomposition of $G - e'$ corresponds to rerooting $\mathcal{T}$ at the node representing e'. It thus makes sense to say that $\mathcal{T}$ is the SPQR-tree of G.

The SPQR-tree $\mathcal{T}$ of a graph G with n vertices and m edges has m Q-nodes and $O(n)$ S-, P-, and R-nodes. Also, the total number of vertices of the skeletons stored at the nodes of $\mathcal{T}$ is $O(n)$. Finally, SPQR-trees can be constructed and handled efficiently. Namely, given a biconnected planar graph G, the SPQR-tree $\mathcal{T}$ of G can be computed in linear time [GM00].

SPQR-trees and Planar Embeddings. Let G be a biconnected graph and let $\mathcal{T}$ be the SPQR-tree of G. Recall that graph G is planar if and only if all its triconnected components are planar, that it, graph G is planar if and only if the skeletons of all the nodes of $\mathcal{T}$ are planar [BDD00].

The SPQR-tree $\mathcal{T}$ can be used to represent all the planar embeddings of G. In fact, suppose that one of the combinatorial embeddings of the skeleton of each node is chosen. A combinatorial embedding of G can be obtained by merging the skeletons of all the adjacent nodes of $\mathcal{T}$ while preserving their embedding.

Observe that:

(i) the skeleton of an S-node admits exactly one combinatorial embedding (in fact, since each vertex has degree two it admits a single rotation scheme);

(ii) the skeleton of a P-node admits as many combinatorial embeddings as the number of permutations of its virtual edges; and

(iii) the skeleton of an R-node, which is triconnected, admits exactly one combinatorial embedding up to a reversal of the adjacency lists of its vertices.

Hence, a combinatorial embedding Γ of G correspond bijectively to planar embeddings of all skeletons of $\mathcal{T}$; the choices are the orderings of the parallel edges in P-nodes and selection of one of the two possible embeddings of the R-node skeletons. When considering rooted SPQR-trees, we assume that the embedding of G is such that the root edge is incident to the outer face, which is equivalent to the parent edge being incident to the outer face in each skeleton.

We remark that in a planar embedding of G, the poles of any node μ of $\mathcal{T}$ are incident to the outer face of $pert(\mu)$; embedding satisfying this property are called *regular* in [BM90]. Hence, in the chapters of this thesis we will only consider embeddings of the pertinent graphs with their poles lying on the same face.

In addition to the applications of this data structure presented in [DT90, DT96a, DT96b], the interested reader might refer, for example, to [GMW05, FGJ+08, ADF+10, ADP11, ADF+12, ACDP13, HN14]; see also Section 4.3 and Chapter 5.

2.4 PQ-trees

A *PQ-tree* $\mathcal{T}$ is a rooted tree whose leaves correspond to the elements of a ground set A. The internal nodes of $\mathcal{T}$ are either P- or Q-nodes. P-nodes allow for an arbitrary reordering of their children, while Q-nodes only allow for a reversal of the given order of their children. PQ-trees can be used to represent all and only the *linear orderings* of their leaves in which subsets of the leaves are required to be consecutive.

Usually, PQ-trees are depicted with circles representing P-nodes and rectangles representing Q-nodes. See Fig. 2.3 for an example.

PQ-trees were introduced by Booth and Lueker [Boo75, BL76] who exploited this data structure to test matrices for the *consecutive-ones property*, to implement a

Figure 2.3: (a) A P-node and (b) a Q-node depicted as a circle and a rectangle, respectively, with *children* (the subtrees $T_1, T_2, \ldots, T_k$) drawn below them.

linear-time *planarity testing* algorithm, and to recognize *interval graphs*. They also proved that construction and intersection of PQ-trees can be performed in linear time by repeatedly applying the nine transformation rules called the *template matchings*.

Let T be a PQ-tree and let $\mathcal{L}(T)$ be the set of leaves of T. We denote by $F(T)$ the set of *linear orders* of $\mathcal{L}(T)$ obtainable by arbitrarily reordering the children of each P-node and by selecting one of the two possible orderings of the children of each Q-node. Also, let $X \subseteq \mathcal{L}(T)$ be a subset of the leaves of T. We say that X is *representable* by T, denoted $X \circledR T$, if there exists a linear ordering $\mathcal{O} \in F(T)$ such that the elements of X appear consecutively in $\mathcal{O}$; otherwise, we say that X is *not representable* by T, denoted $X \overline{\circledR} T$. Further, we denote by $T_\emptyset$ the PQ-tree representing the empty order, that is, $F(T_\emptyset) = \emptyset$, and by T_∞ the PQ-tree representing all the possible permutations of its leaves, that is, T_∞ is a PQ-tree with a unique internal node that is a P-node.

We now illustrate two fundamental operations that are defined on PQ-trees. Namely, the *reduction* and the *intersection* of PQ-trees.

Reduction. Let T be a PQ-tree and let $X \subseteq \mathcal{L}(T)$ be a subset of the leaves of T. The *reduction* of T on X, denoted by $T \bigoplus X$, constructs a new PQ-tree T^* such that either (i) T^* is a PQ-tree with leaves $\mathcal{L}(T^*) = \mathcal{L}(T)$ and such that, for each ordering $\mathcal{O} \in F(T)$ in which the elements of X appear consecutively, it holds that $\mathcal{O} \in F(T^*)$, or (ii) $T^* = T_\emptyset$, if $X \overline{\circledR} T$.

Intersection. Let T' and T'' be two PQ-tree on the same set of leaves. The *intersection* of T' and T'', denoted by $T' \cap T''$, constructs a new PQ-tree T^* such that either (i) T^* is a PQ-tree with leaves $\mathcal{L}(T^*) = \mathcal{L}(T') = \mathcal{L}(T'')$ and such that $F(T^*) = F(T') \cap F(T'')$, or (ii) $T^* = T_\emptyset$, if $F(T') \cap F(T'') = \emptyset$.

Observe that, given a PQ-tree T, there exist a collection $X(T)$ of sets $X_i \subseteq \mathcal{L}(T)$, with $1 \leq i \leq k$, and PQ-trees T_i, with $1 \leq i \leq k$, such that (i) $T_1 = T_\infty$, (ii) $T_{i+1} =$

$T_i \bigoplus X_i$, for $i = 1, \ldots, k-1$, and (iii) $T_k = T$. Hence, the intersection between two PQ-trees T' and T'' can be obtained from T' (resp., from T'') by applying $|X(T'')|$ (resp., $|X(T')|$) reduction steps, one for each set in $X(T'')$ (resp., in $X(T')$).

In addition to the applications of this data structure presented in [Boo75, BL76], the interested reader might refer, for example, to [FCE95b, JLM98, LPW05]; see also Section 4.3 and Section 8.3.

Chapter 3

Planarity of Simultaneous and Clustered Graphs

In this chapter we introduce the two long-standing Graph Drawing open problems which are the main subject of this thesis. Namely, Section 3.1 is devoted to the CLUSTERED PLANARITY (C-PLANARITY for short) problem, while in Section 3.2 we discuss the SIMULTANEOUS EMBEDDING WITH FIXED EDGES (SEFE for short) problem. Further, we present the state of the art about testing this notions of planarity for several classes of clustered graphs and simultaneous graphs.

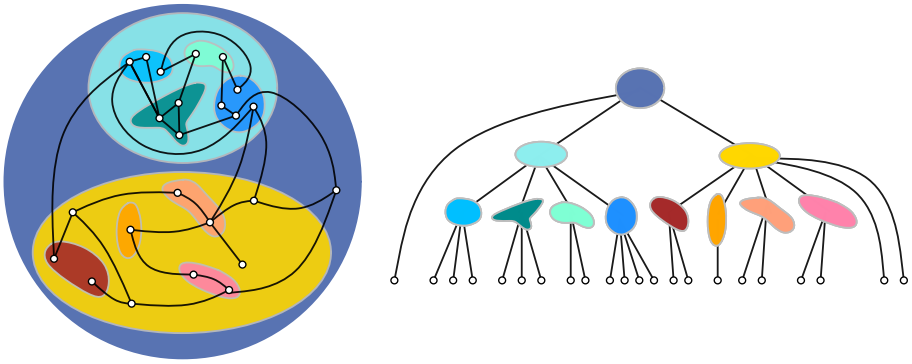
3.1 Clustered Graphs

In this section we give definitions about clustered graphs and review the state of the art of the C-PLANARITY testing problem[1].

Let X be any set of elements. A *cluster* of X is a non empty subset $X' \subseteq X$ of its vertex set X. A *recursive clustering* of X is an assignment of its elements to clusters such that, for any two clusters $X' \subseteq X$ and $X'' \subseteq X$, it holds that either (i) $X' \cap X'' = X'$, or (ii) $X' \cap X'' = X''$, or (iii) $X' \cap X'' = \emptyset$. A recursive clustering of X can be conveniently represented by means of a rooted tree T whose leaves are the elements in X and whose internal nodes represent clusters of X.

A *clustered graph* $C(G, T)$ (*c-graph* for short) is a graph $G(V, E)$ plus a rooted tree T representing a recursive clustering of V. A *cluster* (resp. *recursive clustering*)

[1] We remark that, as clarified in Feng's thesis [Fen97], the term C-PLANARITY was originally meant as a abbreviation for *compound planarity*. Afterwards, the term has commonly been accepted as a short form for CLUSTERED PLANARITY.



(a) (b)

Figure 3.1: (a) A c-planar drawing Γ of a clustered graph $C(G,T)$. (b) The inclusion tree T of $C(G,T)$, where each internal node $\mu \in T$ is represented as a scaled drawing of the region $R(\mu)$ representing μ in Γ.

Figure 3.2: (a) A c-planar drawing of a flat clustered graph and its inclusion tree. (b) A c-planar drawing of a non-flat clustered graph and its inclusion tree.

of $G(V,E)$ is a cluster (resp. recursive clustering) of the vertex set V of G. Given an internal node μ of T we denote by $V(\mu)$ the cluster of G whose vertices are the leaves of the subtree of T rooted at μ. The subgraph of G induced by $V(\mu)$ is denoted as $G(\mu)$. An edge e between a vertex of $V(\mu)$ and a vertex of $V \setminus V(\mu)$ is said to be *incident* on μ. Graph G and tree T are called *underlying graph* and *inclusion tree* (or *cluster hierarchy tree*), respectively. A clustered graph $C(G,T)$ is *flat* if T is a tree of height 2 (that is, removing all the leaves yields a star graph) and *non-flat* otherwise.

See Fig 3.1 for an example of a non-flat c-planar clustered graph and of its inclusion tree.

Graphs coming from many real application scenarios often exhibit semantic affinities among their vertices that allow to group them into clusters. Further, in order to cope with large graphs, clusters may also be artificially introduced with the purpose of making easier the navigation and the exploration of large graphs. In fact, clusters allows for representations of the same graph at different levels of abstraction, where large portions of the graph may be hidden by contracting entire clusters to single vertices (see e.g. [DGK03]). Refer to Fig 3.3.

Let $I = [a,b]$ be a closed interval of the real numbers. Then a *curve* γ is a

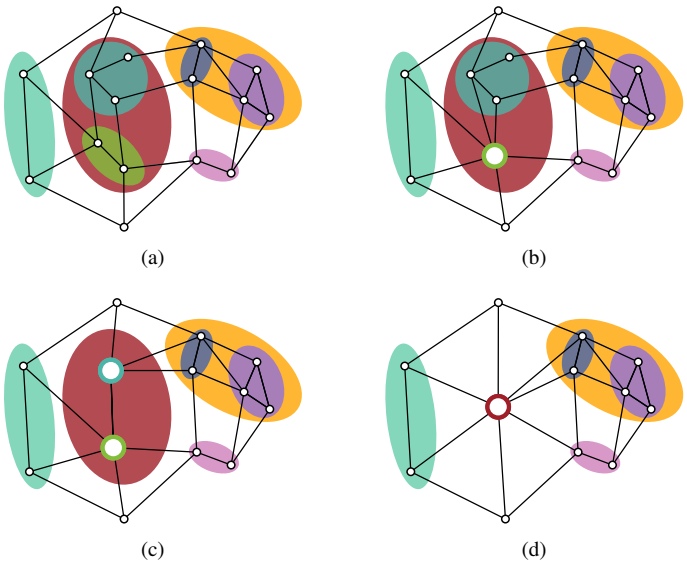


Figure 3.3: (a) A drawing of a clustered graph $C(G,T)$. ((b),(c), and (d)) Drawings of clustered graph $C(G,T)$ at different levels of abstractions, where some clusters are contracted to single vertices which are represented by large circles.

continuous mapping $\gamma : I \to R^2$. A curve γ is *simple*, or a *Jordan arc*, if it is injective, that is, for all $x, y \in I$, it hold that $\gamma(x) = \gamma(y)$ if and only if $x = y$. A curve γ is *closed* if $\gamma(a) = \gamma(b)$. A *simple closed curve*, also called a *Jordan curve*, is a simple curve in which $\gamma(a) = \gamma(b)$. A *region of the plane* (or simply *region*) is subset of R^2 that is open, connected and non-empty. A *simple closed region* is a region bounded by a simple closed curve γ that includes γ, that is, it contains all its boundary points.

A *c-planar* drawing Γ of a clustered graph $C(G,T)$ is a drawing of G together with a drawing of each cluster $\mu \in T$ as a simple closed region $R(\mu)$ such that: 1. $R(\mu)$ encloses all and only the leaves of $T(\mu)$ and the regions representing the internal nodes of $T(\mu)$; 2. no *edge-edge crossing* occurs, the drawing of G contained in $Gamma$ is planar; 3. no *region-region crossing* occurs, that is, $R(\mu) \cap R(\nu) \neq \emptyset$ if and only if ν is an internal node of $T(\mu)$; and 4. no *edge-region crossing* occurs, that is, each edge (u,v) of G intersects the boundary of $R(\mu)$ at most once.

A clustered graph is *c-planar* if it admits a c-planar drawing. The CLUSTERED

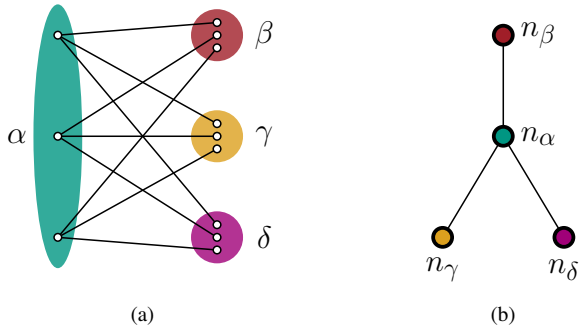


Figure 3.4: (a) A clustered graph $C(G,T)$ with four clusters α, β, γ, and δ that is not c-planar, whose underlying graph is a set of stars. (b) The graph of clusters of $C(G,T)$, containing a vertex n_μ for each cluster $\mu \in T$.

PLANARITY problem (C-PLANARITY for short), introduced by Feng, Cohen, and Eades [FCE95b, FCE95a], asks whether a clustered graph is c-planar.

Let $C(G,T)$ be a flat clustered graph and let H be the *graph of clusters* of $C(G,T)$, that is, the simple graph obtained from G by contracting the subgraph $G(\mu)$ induced by the vertices of cluster μ to a single vertex n_μ and by removing loops and multiple edges. Given a clustered graph $C(G,T)$ an obvious necessary condition for the c-planarity of $C(G,T)$ is the planarity of its underlying graph G. Further, given a flat clustered $C(G,T)$ a necessary condition for the c-planarity of $C(G,T)$ is the planarity of its graph of clusters. However, this conditions are not also sufficient and the consequences on the problem due to the requirement of not having edge-region and region-region crossings are not yet fully understood. See Fig 3.4 for an example of a flat non c-planar clustered graph whose underlying graph and whose graph of clusters are planar.

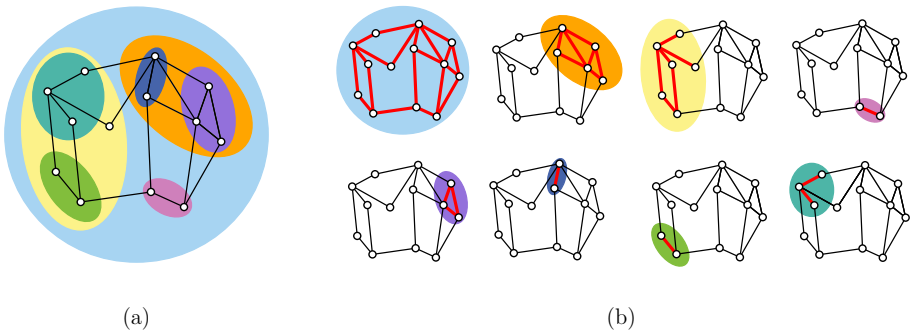
The complexity of the C-PLANARITY problem is still unknown, both in the cases in which the embedding of the underlying graph is fixed or can vary. In the next subsection we review the state of the art on testing c-planarity for restricted classes of clustered graphs.

Determining the complexity of testing whether a clustered graph admits a c-planar drawing is a long-standing open problem in the Graph Drawing research area (see [CB05] for a survey).

Testing C-PLANARITY

In the last decades the C-PLANARITY problem has been deeply studied. While the complexity of deciding if a clustered graph is c-planar is still an open problem in the general case, polynomial-time algorithms have been proposed to test c-planarity and produce c-planar drawings under several kinds of (possibly combined) restrictions. In the following we review several classes of clustered graph for which the C-PLANARITY problem is known to be testable in polynomial-time. Results are classified according to the most adopted restrictions; we omit repetitions when multiple restrictions are be applied.

Constraints on the Connectivity of Clusters. A cluster μ is *connected*, if the graph $G(\mu)$ induced by the vertices of $V(\mu)$ is connected, and *non-connected*, otherwise. A clustered graph is *c-connected* if for each node μ of T we have that μ is connected. For example, the clustered graph $C(G,T)$ in Fig. 3.5(a) is c-connected; in fact, as shown in Fig. 3.5(b), each cluster induces a connected subgraph of G.



(a) (b)

Figure 3.5: (a) A c-connected clustered graph $C(G,T)$. (b) For each internal node $\mu \in T$, the edges of graph $G(\mu)$ are drawn as thick red segments.

An important reference point in the literature on clustered planarity is the work by Feng *et al.* [FCE95b, FCE95a]. In these papers the terminology that is currently adopted is defined and an algorithm to test the c-planarity of a c-connected clustered graph is provided. The algorithm is based on PQ-trees (refer to Section 2.4 for an introduction on this data structure) and runs in quadratic time. In their original work, Feng *et al.* also provided the following elegant characterization for c-planar clustered graphs.

Theorem 3.1 ([FCE95b]) *A clustered graph $C(G,T)$ is c-planar if and only if it is a sub-clustered graph of a c-connected and c-planar clustered graph.*

In a different context, Lengauer [Len89] gave an algorithm for testing planarity of graphs defined in a hierarchical fashion. Namely, in that case that the clustered graph is specified in terms of a set of graph patterns and in terms of their composition. This algorithm can be interpreted as a C-PLANARITY testing for c-connected clustered graph. The time complexity of the algorithm is linear in the size of the input. However, the input size of Lengauer's algorithm can be quadratic in the size of the represented clustered graph.

The first linear-time algorithm for testing c-planarity of c-connected clustered graphs was presented by Dahlhaus [Dah98]. The algorithm is based on a decomposition of G into its biconnected and triconnected components, a weight of each cluster proportional to its size, and the characterization of c-planar embeddings presented in [FCE95b]. The testing algorithm incrementally constructs of a certain planar embedding and finally checks whether this embedding is c-planar. Shortly after, a new linear-time algorithm was given by Cortese *et al.* [CDF$^+$08] for testing c-planarity of the class of c-connected clustered graph. The algorithm shares many aspects with that presented by Dahlhaus [Dah98] and is based on exploiting both the BC-tree data structure and the SPQR-tree data structure (refer to Section 2.2 and to Section 2.3 for an introduction on these data structures).

The general C-PLANARITY testing problem, for non-connected clustered graphs, is still open. In the following we review the steps that were done in the direction of settling the general question, where several properties related to the connectivity of clusters have been exploited.

A clustered graph is *completely connected* if for each non-root inner node μ of T both $G(\mu)$ and $G \setminus G(\mu)$ are connected. Completely connected clustered graphs have been studied by Cornelsen and Wagner [CW06], who proved that a completely connected clustered graph is c-planar if and only if its underlying graph is planar.

A clustered graph is *almost connected* if it holds that either 1. there exists a unique path p starting from the root of T such that each internal node μ of T that is non-connected lies in p, or 2. for each internal node μ of T that is non-connected, its parent and all its siblings in T are connected.

Gutwenger *et al.* [GJL$^+$02, GJL$^+$03] presented a $O(n^2)$-time algorithm to test if an n-vertex almost connected clustered graph is c-planar.

A cluster μ is *extrovert* if its parent ν in T is connected and each connected component of μ has a edge that is incident to a cluster which is not a descendant of μ. A clustered graph is *extrovert* if all its clusters are connected or extrovert. Extrovert clustered graphs were introduced by Goodrich *et al.* [GLS05] who also gave a cubic-

time algorithm to test c-planarity of this class of clustered graphs and an embedding algorithm with the same time complexity.

Flat Clustered Graphs with Two Clusters. Biedl *et al.*[BKM98] study planar graphs where each vertex is assigned to one of two disjoint sets. They provide a linear-time algorithm to test if one of such graphs has a planar drawing such that the vertices of the two classes are separated by an horizontal line (*y-monotone HH-drawing*). Clearly, this algorithm can be interpreted as a c-planarity testing of a graph with exactly two clusters (excluding the root) both at the same level. Shortly after, Hong and Nagamochi [HN14] provided a linear-time algorithm to test c-planarity of a clustered graph with arbitrary underlying graph whose vertices are assigned to two clusters (excluding the root), based on the equivalence between the C-PLANARITY in this setting and the PARTITIONED T-COHERENT 2-PAGE BOOK EMBEDDING (PBE-2) (see Section 3.2 for a discussion on this problem), for which they provided a linear-time testing algorithm based on SPQR-trees. More recently the same authors presented a simpler linear-time algorithm to test the PBE-2 problem, based on reducing the problem to a planarity test.

Flat Clustered Graphs with Fixed Embedding of the Underlying Graph. A *candidate saturating edge* in a clustered graph $C(G,T)$ is an edge whose addition to G decreases the number of connected components of some cluster of $C(G,T)$. Given an embedded flat clustered graph, two candidate saturating edges have a *conflict* if adding both of them to G causes a crossing. Di Battista and Frati [DF09] present a characterization for *single-conflict embedded flat clustered graphs*, that are embedded clustered graphs such that (i) the cluster hierarchy is flat and (ii) each candidate saturating edge has a conflict with at most one other candidate saturating edge. As a consequence of such a more general result, they provide a linear-time algorithm for embedded flat clustered graphs with at most five vertices per face.

Recently, Chimani *et al.* [CDFK14] have shown a cubic-time algorithm for testing c-planarity of embedded flat clustered graphs with at most two vertices per cluster on each face. They also introduce a generalization of the C-PLANARITY problem for embedded flat clustered graphs, called PLANAR SET OF SPANNING TREES IN TOPOLOGICAL MULTIGRAPHS (PSSTTM for short), defined as follows: Given a non-planar topological multigraph G with k connected components $C_1, \ldots, C_k$, the PSSTTM problem asks whether there exists spanning trees $S_1, \ldots, S_k$ of $C_1, \ldots, C_k$ such that no two edges in $\bigcup_i S_i$ cross. The PSSTTM problem is NP-hard, even if $k = 1$ [KLN91].

Small Number of Components per Cluster. Jelínek *et al.* [JJKL08] showed that the C-PLANARITY problem can be tested in polynomial time, if the embedding is fixed and each cluster induces a subgraph with at most two connected components. A somewhat more general result has recently been proved by Bläsius and Rutter [BR14], who gave an efficient algorithm for testing c-planarity in the variable embedding setting in the case in which every cluster and every co-cluster induces at most two connected components. Their algorithm is based on exploiting the machinery of Simultaneous PQ-Ordering; see also [BR13] for further details on this topic.

Focusing on Particular Families of Underlying Graphs. In [CDPP05] Cortese *et al.* focus on 3-cluster cycles, which are flat clustered graphs such that the underlying graph is a simple cycle whose vertices are partitioned into 3 clusters. They show that deciding C-PLANARITY for such instances and computing a c-planar drawing can be done in linear time. Further, they give an elegant characterization of 3-cluster cycles in terms of formal grammars.

Jelínková *et al.* [JKK$^+$07] study the C-PLANARITY testing problem when all clusters are small. Their main result is an $O(|\mathcal{C}|^3 + n)$-time algorithm for clusters of size at most three on a cycle, where $\mathcal{C}$ is the set of clusters. Such a result is generalized to a special class of Eulerian graphs that can be obtained from a vertex-3-connected planar graph of fixed size by cloning and subdividing edges.

Limiting the Number of Inter-Cluster Edges. Polynomial-time algorithm have been devised by posing restriction on the edges crossing the boundary of clusters. In [JSTV08] Jelínek *et al.* give a linear-time algorithm for clusters with at most four outgoing edges. This result is based on simulating the behavior of clusters with special graphs, no matter whether the subgraph induced by each cluster is connected or not. A slightly better result has been recently proved by Bläsius and Rutter [BR14] who have shown a polynomial-time algorithm for C-PLANARITY when every cluster has at most five outgoing edges.

Constraining the Arrangement of Clusters. Let $C(G, T)$ be a flat clustered graph and let H be the graph of clusters of $C(G, T)$. As already observed, the planarity of H is a necessary condition for the c-planarity of $C(G, T)$. Also, let Γ_H be a planar drawing of H in which each vertex is drawn by a closed disk and each edge is drawn by a pipe connecting the discs corresponding its endpoints. Cortese *et al.* [CDPP09] study a problem in the framework of clustered planarity for highly non-connected flat clustered graphs, defined as follows. Given a flat clustered graph $C(G, T)$ and a planar drawing Γ_H of the graph of clusters H of $C(G, T)$, does there exists a planar

drawing Γ_G of G such that, for each cluster μ of T, graph $G(\mu)$ is drawn inside the disk representing vertex n_μ in Γ_H and each inter-cluster edge (u, v) with $u \in \mu$ and $v \in \nu$ is drawn as a curve completely contained inside the pipe connecting vertex n_μ and vertex n_ν in Γ_H? They show that this problem can be solved in cubic-time if G is a cycle. Recently, Chang *et al.* [CEX15] improved on the result of [CDPP09] by presenting an $O(n \log n)$-time algorithm to determine whether a closed walk of length n in a simple plane graph is weakly simple[2].

Finally, in [CDPP05] Cortese *et al.* studied the class of non-connected clustered graphs such that the underlying graph is a cycle and the clusters at the same level of T also form a cycle, where two clusters are considered adjacent if they are incident to the same edge. They show that the C-PLANARITY testing and embedding problem is linear for this class of clustered graphs.

3.2 Simultaneous Graphs

In this section we give definitions about simultaneous graphs and review the state of the art of the SEFE testing problem.

A *simultaneous graph* is a triple $G = (V, E, \psi)$ where V is a set of vertices, $E \subseteq V \times V$ is a set of edges, and $\psi : V \to \mathcal{P}(\{1, \ldots, k\})$ is a function that associates each edge of the simultaneous graph with a set of integers with values up to k. Given k graphs $G_1 = (V, E_1), \ldots, G_k = (V, E_k)$ defined on the same vertex set V, a simultaneous graph $G = (V, E, \psi)$ can be defined such that $E = \bigcup_i^k E_i$ and, for each edges $e \in E$, $\psi(e) = \{i : e \in E_i\}$. Simultaneous graphs can be used to represent multiple sets of relationships over the same set of entities or to show the evolution of a single graph whose edge-set changes over time. In the chapters of this thesis dealing with simultaneous graphs, with the exception of Chapter 11, we will be mostly interested in simultaneous graphs that are the union of a small number of planar graphs. This will allow us to describe a simultaneous graph explicitly by means of its constituent graphs and to drop the notation exploiting function ψ. For example, we will denote by $\langle G_1, G_2 \rangle$, by $\langle G_1, G_2, G_3 \rangle$, and by $\langle G_i(V, E_i) \rangle_{i=1}^k$ simultaneous graphs composed of 2, 3, and k planar graphs, respectively. In Chapter 11, however, where simultaneous graphs described as a stream of edges are studied, we will exploit a notation similar to the one introduced above.

Clearly, even though each graph G_i, with $i = 1, 2, \ldots, k$, is planar, the simultaneous graph G associated with these graphs might – in general – be non planar. In the following, we discuss a notion of planarity for simultaneous graphs which aims

[2] A closed curve in the plane is *weakly simple* if it is the limit in the Fréchet metric of a sequence of simple closed curves.

at (i) clearly displaying each of the graph composing a simultaneous graph and at (ii) helping the user maintain his/her mental map when exploring the structure of the simultaneous graph.

Let $G_1 = (V, E_1), \ldots, G_k = (V, E_k)$ be k planar graphs on the same set V of vertices. A *simultaneous embedding with fixed edges* (SEFE for short) of $G_1, \ldots, G_k$ consists of k planar drawings $\Gamma_1, \ldots, \Gamma_k$ of $G_1, \ldots, G_k$, respectively, such that: (i) each vertex $v \in V$ is mapped to the same point in every drawing Γ_i and (ii) each edge that is common to more than one graph is represented by the same simple curve in the drawings of all such graphs. The SIMULTANEOUS EMBEDDING WITH FIXED EDGES (SEFE for short) problem asks whether k input graphs $G_1, \ldots, G_k$ admit a SEFE [EK05]. We denote an instance of the SEFE problem as $\langle G_1 = (V, E_1), \ldots, G_k = (V, E_k)\rangle$ or, using a more concise notation, as $\langle G_i(V, E_i)\rangle_{i=1}^{k}$. Given an instance $\langle G_i(V, E_i)\rangle_{i=1}^{k}$ of SEFE, the graph $G_{\cap}^{ij} = (V, E_i \cap E_j)$ is the *common graph* of graphs G_i and G_j, with $1 \leq i < j \leq k$. In this thesis we will be mostly concerned with the cases $k = 2$ and $k = 3$. To easy the description, we will consider the edge-sets E_1, E_2, and E_3 of each graph assigned a color between red, blue, and green. Also, for the sake of consistency, we will denote graphs $G_1 = (V, E_1)$, $G_2 = (V, E_2)$, and $G_3 = (V, E_3)$ as the red graph, as the blue graph, and as the green graph, respectively. Furthermore, we will explicitly refer to the variant of the problem in which $k = 2$ graphs G_1 and G_2 are considered, as the SEFE-2 problem. For example, Fig. 3.6 shows a negative instance $\langle G_1, G_2\rangle$ of SEFE-2.

In the following we introduce some notable variants of the SEFE problem.

Let $\langle G_i(V, E_i)\rangle_{i=1}^{k}$ be an instance of SEFE such that for each two graphs G_i and G_j, with $i \neq j$, it holds that $G_i \cap G_j = G_{\cap}$, where $G_{\cap} = \bigcap_{l=1}^{k} G_l$. We say that the input graphs have *sunflower intersection*, that is, the intersection graph is the same for each pair of input graphs. We call SUNFLOWER SEFE problem the restriction of the SEFE problem to this class of instances.

Let $\langle G_1, G_2\rangle$ be an instance of SEFE-2 such that the *common graph* $G_{\cap} = (V, E_1 \cap E_2)$ is connected. We call C-SEFE-2 problem the restriction of the SEFE-2 problem to such instances. Observe that, any instance of SEFE-2, and hence C-SEFE-2, is also an instance of SUNFLOWER SEFE.

Given k planar graphs $G_1 = (V, E_1), \ldots, G_k = (V, E_k)$ such that $E_i \cap E_j = \emptyset$, with $1 \leq i < j \leq k$, a *k-page book-embedding* of graphs $G_1 = (V, E_1), \ldots, G_k = (V, E_k)$ consists of a linear ordering $\mathcal{O}$ of the vertices of V such that for each set E_i, with $1 \leq i \leq k$, there exist no two edges $e', e'' \in E_i$ whose endvertices alternate in $\mathcal{O}$. The PARTITIONED T-COHERENT k-PAGE BOOK EMBEDDING problem (PTBE-k for short) is defined as follows. Given a set V of vertices, a tree T whose leaves $\mathcal{L}(T)$ are the elements of V, and a collection of edge-sets $E_i \subseteq V \times V$, for

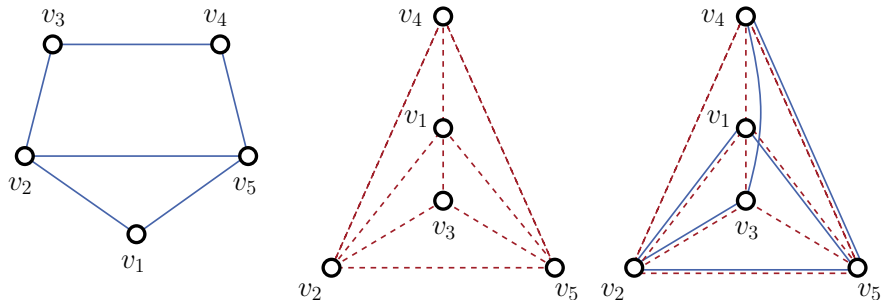


Figure 3.6: Graphs G_1 (a) and G_2 (b) do not admit a SEFE-2. (c) Since G_1 is tricon-nected, vertices v_4 and v_3 have to lie on different sides of cycle $\mathcal{C} = (v_1, v_2, v_5, v_1)$ in any of the two combinatorial embeddings of G_1. Hence, edge (v_3, v_4) of G_2 cannot be added without introducing crossings with the edges of $\mathcal{C}$.

$i = 1, 2, \ldots, k$, such that $E_i \cap E_j = \emptyset$, with $1 \leq i < j \leq k$, the PTBE-k asks whether there exists an ordering $\mathcal{O}$ of the elements of V such that (i) the ordering $\mathcal{O}$ is represented by T and (ii) the endvertices of any two edges belonging to the same set E_i do not alternate in $\mathcal{O}$. We denote an instance of the PTBE-k problem by $\langle T, E_1, \ldots, E_k \rangle$. Intuitively, the problem aims at placing the vertices along the *spine* of a book in such a way that (i) the placement is consistent with tree T and (ii) it is possible to draw the edges of each set on a page of the book without creating any crossing. We remark that the "connected" version of SEFE-2 is equivalent to problem PARTITIONED T-COHERENT k-PAGE BOOK EMBEDDING (PTBE-k for short) [ADF+12] for $k = 2$. Furthermore, we refer to the version of the PTBE-k problem in which tree T is a star graph as the PARTITIONED k-PAGE BOOK EMBEDDING (PBE-k for short) problem [HN09, ADD12, HN14]. We denote an instance of the PARTITIONED k-PAGE BOOK EMBEDDING problem simply by $\langle V, E_1, \ldots, E_k \rangle$. See Fig. 3.7 for examples of instances of these problems.

The SEFE problem can be studied both in terms of embeddings and in terms of drawings, since edges can be represented by arbitrary curves without geometric restrictions, and since Jünger and Schulz [JS09] proved that two graphs G_1 and G_2 with common graph $G_\cap$ have a SEFE-2 if and only if there exists a planar embedding Γ_1 of G_1 and a planar embedding Γ_2 of G_2 inducing the same embedding of $G_\cap$. This condition extends to more than two graphs in the sunflower intersection setting.

Let S be a subgraph of a planar graph G and let $\mathcal{E}$ be a planar embedding of G.

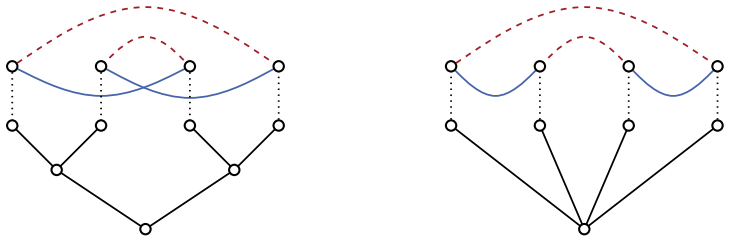


Figure 3.7: (a) A negative instance $\langle T, E_1, E_2 \rangle$ of PTBE-k and (b) a positive instance $\langle G_1, G_2 \rangle$ of PTBE-2 where the $V = \mathcal{L}(T)$, $E(G_1) = E_1$, and $E(G_2) = E_2$. The constraints imposed by tree T do not allow for an ordering $\mathcal{O}$ of its leaves along the spine that yields a crossing-free drawing on both pages.

We denote by $\mathcal{E}|_S$ the embedding of S induced by $\mathcal{E}$. Similarly, let S be a subgraph of a (not necessarily planar) graph G and let Γ be a drawing of G. We denote by $\Gamma|_S$ the drawing of S induced by Γ.

Jünger and Schulz [JS09] showed that the SEFE-2 problem can be equivalently stated in terms of embeddings. Namely, two graphs $G_1 = (V, E_1)$ and $G_2 = (V, E_2)$ whose intersection graph $G_\cap = (V, E_1 \cap E_2)$ is connected admit a SEFE if and only if there exist planar embeddings $\mathcal{E}_1$ and $\mathcal{E}_2$ of G_1 and G_2, respectively, such that $\mathcal{E}_1|_{G_\cap} = \mathcal{E}_2|_{G_\cap}$ holds, that is, the two embeddings $\mathcal{E}_1$ and $\mathcal{E}_2$ coincide when restricted to the intersection graph. Jünger and Schulz refer to embeddings satisfying this property as *compatible embeddings*. The following theorem formalizes this result.

Theorem 3.2 ([JS09]) *Let G_1 and G_2 be two planar graphs. G_1 and G_2 admit a SEFE-2 if and only if there exist a pair of compatible embeddings $\mathcal{E}_1$ and $\mathcal{E}_2$ of G_1 and G_2, respectively.*

Determining the complexity of testing whether an instance $\langle G_1, G_2 \rangle$ of SEFE-2 (that is, a simultaneous graph composed of only two planar graphs) admits a SEFE-2 is a fascinating open problem in Graph Drawing. We refer the interested reader to [BKR13b] for a comprehensive survey on this topic.

Testing SEFE

In recent years the SEFE problem has been deeply studied. Deciding whether k graphs admit a SEFE is an NP-complete problem [GJP+06], with $k \geq 3$. This is

also true when every pair of graphs shares the same common graph (SUNFLOWER SEFE) [Sch13]; see also Chapter 8. On the other hand, if the embedding of the input graphs is fixed, SEFE becomes polynomial-time solvable for $k = 3$, but remains NP-complete for $k \geq 14$ [ADF13]. Also, Hoske [Hos12] proved NP-completeness of PTBE-k, that is the restriction of SUNFLOWER SEFE to instances whose common graph $G_\cap$ is a star, for k unbounded; see also Chapter 8. Nonetheless, the complexity of testing SEFE for two graphs, hereinafter denoted as SEFE-2, is still open in the general setting, even in its "connected" version PTBE-2 [ADF$^+$12]. In a recent work [Sch13] by Marcus Schaefer polynomial-time reductions from several well-known open problems in Graph Drawing, including C-PLANARITY, to the SEFE-2 problem have been presented. This results further motivate the strong interest in settling the question regarding the computational complexity of this problem. In the following we review several classes of simultaneous graphs for which the SEFE-2 problem is known to be testable in polynomial-time.

Constraints on the Connectivity of the Instance. Haeupler *et al.* [HJL13] and Angelini *et al* [ADF$^+$12] independently show a linear-time algorithm to solve SEFE for the case that the common graph is biconnected. The algorithm in [HJL13] is an extension of the planarity testing algorithm by Haeupler and Tarjan [HT08] that is bases on the PQ-tree data structure; while the algorithm in [ADF$^+$12] aims at finding an embedding of the common graph such that the exclusive edges of G_1 and of G_2 can be added without braking planarity and is based on the SPQR-tree data structure. This algorithms extend to the case of k planar graphs with sunflower intersection. Recently, Schaefer [Sch13] extended this result by providing a polynomial-time algorithm for the case in which the common graph consists of the disjoint union of biconnected components and subcubic components, based on an algebraic approach derived from the Hanani-Tutte theorem [Cho34, Tut70]. Further, in [Sch13] an algorithm for the case in which at least one of G_1 or G_2 is the disjoint union of subdivisions of 3-connected graphs is provided.

Bläsius and Rutter [BR13] give a quadratic-time algorithm to test SEFE-2 of two biconnected planar graphs G_1 and G_2 whose common graph is connected. They formulated the problem in the framework of the SIMULTANEOUS PQ-ORDERING problem that is defined as follows. Let $D = (N, A)$ be a DAG with nodes $N = T_1, \ldots, T_k$, where T_i is an unrooted PQ-tree, and arcs (T_i, T_j) equipped with an injective map $\psi : \mathcal{L}(T_i) \to \mathcal{L}(T_j)$. The SIMULTANEOUS PQ-ORDERING problem asks whether there are orders $\mathcal{O}_1, \ldots, \mathcal{O}_k$ of the leaf-sets $\mathcal{L}(T_1), \ldots, \mathcal{L}(T_k)$ of the PQ-trees $T_1, \ldots, T_k$ that are chosen consistently with respect to the relationship represented by the arcs in A. This result can be slightly extended to the case in which graphs G_1 and G_2 contain

cut-vertices incident to at most two non-trivial blocks (namely, blocks not consisting of a single edge), that includes the special case in which both graphs have maximum degree 5.

Fixing the Embedding of the Input Graphs. Angelini *et al.* [ADF+10] study the following problem: Given a planar graph G and a planar embedding of a subgraph of G, can such an embedding be extended to a planar embedding of the entire graph G? They provide a linear-time algorithm for this problem which immediately yield to solve in linear time the SEFE-2 problem if one of G_1 and G_2 has a fixed embedding.

Bläsius and Rutter [BKR13a] give a linear-time algorithm for solving SEFE-2 if the common graph is a set of disjoint cycles, by introducing the CC-tree data structure. CC-trees can be used to represent all embeddings of a set of disjoint cycles that can be induced by an embedding of a graph containing them as a subgraph. Furthermore, they can extend this result to a quadratic-time algorithm for the case where the common graph consists of arbitrary connected components, each with a fixed planar embedding. Both results extend to the case of k planar graphs with sunflower intersection.

Focusing on Particular Families of Graphs. Fowler *et al.* [FGJ+08] present an algorithm for solving the SEFE-2 problem for a planar graph and a pseudoforest (a graph with at most one cycle). This result is achieved by reducing to the following embedding problem: Given a planar graph G, a cycle C of G, and a partitioning $P = \{P_1, \ldots, P_k\}$ of the remaining vertices of G, does there exist a planar embedding in which the induced subgraph on each vertex partition P_i of $G \setminus C$ is contained entirely inside or outside C? The algorithm for solving is based on the SPQR-tree data structure and has linear running time.

Angelini *et al.* [ADF+12] show the equivalence between SEFE-2 and the PARTITIONED T-COHERENT 2-PAGE BOOK EMBEDDING problem. Based on such an equivalence and on a linear-time algorithm by Hong and Nagamochi for testing the PARTITIONED k-PAGE BOOK EMBEDDING problem [HN09, HN14], they derive a linear-time algorithm for SEFE-2 when the common graph $G_\cap$ is a star.

Some polynomial-time tractable instances have been identified when the degree of the common graph is small. Schaefer [Sch13] shows a polynomial-time algorithm for SEFE-2 when the common graph is subcubic, based on Hanani-Tutte-style methods; while Hoske [Hos12] gives a simple quadratic-time algorithm in the case in which the common graph is a binary tree, based on reducing the problem to the satisfiability of a 2-SAT formula.

Part II

Clustered Planarity

Chapter 4

Relaxing the Constraints of Clustered Planarity

In order to shed light on the C-PLANARITY problem, in this chapter[1] we consider a relaxed version of it, where some kinds of crossings (either edge-edge, edge-region, or region-region) are allowed even if the underlying graph is planar. We investigate the relationships among the minimum number of edge-edge, edge-region, and region-region crossings for drawings of the same clustered graph. Also, we consider drawings in which only crossings of one kind are admitted. In this setting, we prove that drawings with only edge-edge or with only edge-region crossings always exist, while drawings with only region-region crossings may not. Further, we provide upper and lower bounds for the number of such crossings. Finally, we give a polynomial-time algorithm to test whether a drawing with only region-region crossings exists for biconnected graphs, hence identifying a first non-trivial necessary condition for C-PLANARITY that can be tested in polynomial time for a noticeable class of graphs.

4.1 Introduction

C-PLANARITY is a classical Graph Drawing topic (see Chapter 3 for an introduction and for references on this topic). A *clustered graph* $C(G, T)$ consists of a graph G and of a rooted tree T whose leaves are the vertices of G. Such a structure is used to enrich the vertices of the graph with hierarchical information. In fact, each internal node μ of T represents the subset, called *cluster*, of the vertices of G that are the leaves of

[1] The contents of this chapter are a joint work with Patrizio Angelini, Giuseppe Di Battista, Fabrizio Frati, Maurizio Patrignani, and Vincenzo Roselli published in a journal [ADD+15].

the subtree of T rooted at μ. Tree T, which defines the inclusion relationships among clusters, is called *inclusion tree*, while G is the *underlying graph* of $C(G,T)$.

In a *drawing* of a clustered graph $C(G,T)$ vertices and edges of G are drawn as points and open curves, respectively, and each node μ of T is represented by a simple closed region $R(\mu)$ containing exactly the vertices of μ. Also, if μ is a descendant of a node ν, then $R(\nu)$ contains $R(\mu)$.

A drawing of clustered graph $C(G,T)$ can have three types of crossings. *Edge-edge crossings* are crossings between edges of G. Algorithms to produce drawings allowing edge-edge crossings have already been proposed (see, for example, [DDM01] and Fig. 4.1(a)). Two kinds of crossings involve regions, instead. Consider an edge e of G and a node μ of T. If e intersects the boundary of $R(\mu)$ only once, this is not considered as a crossing since there is no way of connecting the endpoints of e without intersecting the boundary of $R(\mu)$. On the contrary, if e intersects the boundary of $R(\mu)$ more than once, we have *edge-region crossings*. An example of this kind of crossings is provided by Fig. 4.1(b), where edge (u,w) traverses $R(\mu)$ and edge (u,v) exits and enters $R(\nu)$. Finally, consider two nodes μ and ν of T; if the boundary of $R(\mu)$ intersects the boundary of $R(\nu)$, we have a *region-region crossing* (see Fig. 4.1(c) for an example). Clearly, a drawing of a clustered graph is c-planar if it does not have any edge-edge, edge-region, or region-region crossing.

The huge body of research on clusterd planarity can be read as a collection of polynomial-time testable sufficient conditions for C-PLANARITY. In contrast, the planarity of the underlying graph is the only polynomial-time testable necessary condition that has been found so far for C-PLANARITY in the general case. Such a condition, however, is not sufficient and the consequences on the problem due to the requirement of not having edge-region and region-region crossings are not yet fully understood.

Other known necessary conditions are either trivial (i.e., satisfied by all clustered graphs) or of unknown complexity as the original problem is. An example of the first kind is the existence of a c-planar clustered graph obtained by splitting some cluster into sibling clusters [AFP09]. An example of the second kind, which is also a sufficient condition, is the existence of a set of edges that, if added to the underlying graph, make the clustered graph c-connected and c-planar [FCE95b].

In this chapter we study a relaxed model of C-PLANARITY. Namely, we study $\langle \alpha, \beta, \gamma \rangle$-drawings of clustered graphs. In an $\langle \alpha, \beta, \gamma \rangle$-drawing the number of edge-edge, edge-region, and region-region crossings is equal to α, β, and γ, respectively. Figs. 4.1(a), 4.1(b), and 4.1(c) show examples of a $\langle 3,0,0 \rangle$-drawing, a $\langle 0,2,0 \rangle$-drawing, and a $\langle 0,0,1 \rangle$-drawing, respectively. Notice that this model provides a generalization of C-PLANARITY, as the traditional c-planar drawing is a special case of an $\langle \alpha, \beta, \gamma \rangle$-drawing where $\alpha = \beta = \gamma = 0$. Hence, we can say that the existence

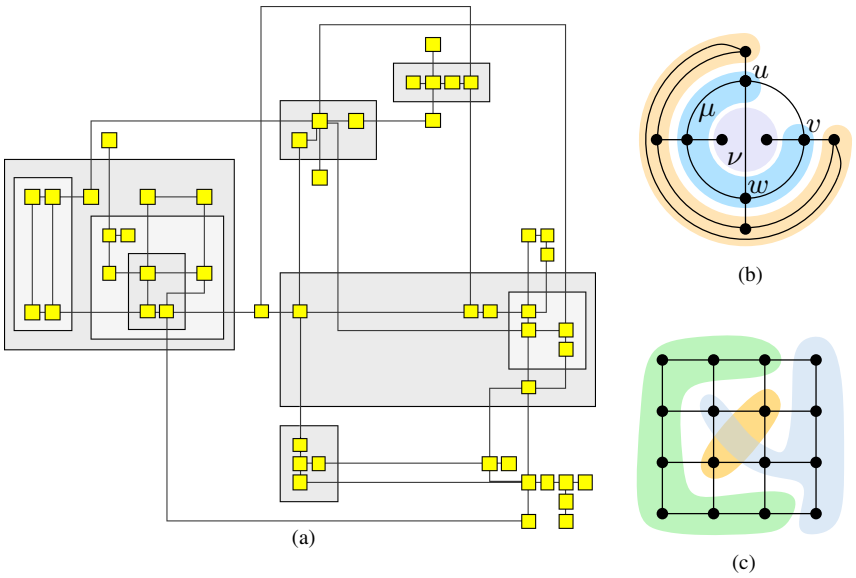


Figure 4.1: Examples of crossings in drawings of clustered graphs. (a) A drawing obtained with the planarization algorithm described in [DDM01] and containing three edge-edge crossings. (b) A drawing with two edge-region crossings. (c) A drawing with a region-region crossing.

of an $\langle \alpha, \beta, \gamma \rangle$-drawing, for some values of α, β, and γ, is a necessary condition for C-PLANARITY.

In our study we focus on clustered graphs whose underlying graph is planar. We mainly concentrate on the existence of drawings in which only one type of crossings is allowed. We call these drawings $\langle \infty, 0, 0 \rangle$-, $\langle 0, \infty, 0 \rangle$-, and $\langle 0, 0, \infty \rangle$-drawings, respectively. Our investigation uncovers that allowing different types of crossings has a different impact on the existence of drawings of clustered graphs (see Fig. 4.2). In particular, we prove that, while every clustered graph admits an $\langle \infty, 0, 0 \rangle$-drawing (even if its underlying graph is not planar) and a $\langle 0, \infty, 0 \rangle$-drawing (only if its underlying graph is planar), there exist clustered graphs not admitting any $\langle 0, 0, \infty \rangle$-drawing. Further, we provide a polynomial-time testing algorithm to decide whether a biconnected clustered graph admits a $\langle 0, 0, \infty \rangle$-drawing. From this fact we conclude that the existence of such a drawing is the first non-trivial necessary condition for the

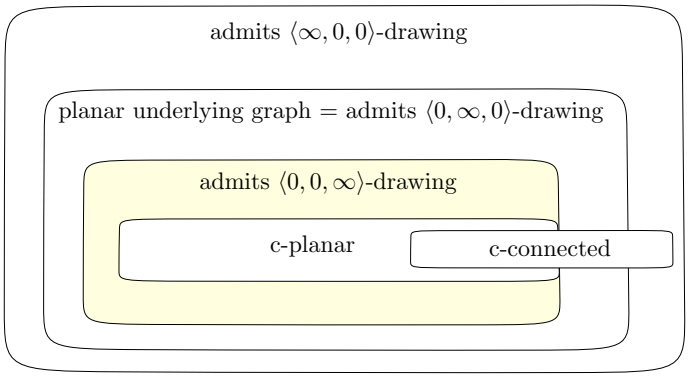


Figure 4.2: Containment relationships among instances of clustered planarity. The existence of a $\langle 0, 0, \infty\rangle$-drawing is a necessary condition for C-PLANARITY. Note that any $\langle 0, 0, \infty\rangle$-drawing of a c-connected clustered graph $C(G, T)$ can be suitably modified to obtain a c-planar drawing of $C(G, T)$.

C-PLANARITY of clustered graphs that can be tested efficiently. This allows us to further restrict the search for c-planar instances with respect to the obvious condition that the underlying graph is planar.

Also, we investigate the relationships among the minimum number of edge-edge, edge-region, and region-region crossings for drawings of the same clustered graph, showing that, in most of the cases, the fact that a clustered graph admits a drawing with few crossings of one type does not imply that such a clustered graph admits a drawing with few crossings of another type.

Finally, we show that minimizing the sum $\alpha + \beta + \gamma$ in a $\langle\alpha, \beta, \gamma\rangle$-drawing of a clustered graph is an NP-complete problem. Since in our construction it is possible to replace each crossing of any type with a crossing of a different type, this implies that the problems of minimizing crossings in $\langle\infty, 0, 0\rangle$-, $\langle 0, \infty, 0\rangle$-, and $\langle 0, 0, \infty\rangle$-drawings are also NP-complete. However, for the first two types of drawings we can prove NP-completeness even for simpler classes of clustered graphs.

We remark that drawings of clustered graphs where a few intersections are admitted may meet the requirements of many typical Graph Drawing applications, and that their employment is encouraged by the fact that the class of c-planar instances might be too small to be relevant for some application contexts.

More in detail, we present the following results (recall that we assume the necessary condition that the underlying graph is planar to be always satisfied):

c-c	flat	$\langle \alpha, 0, 0 \rangle$		$\langle 0, \beta, 0 \rangle$		$\langle 0, 0, \gamma \rangle$	
		α UB	α LB	β UB	β LB	γ UB	γ LB
NO	NO	$O(n^2)$ Th.4.1	$\Omega(n^2)$	$O(n^3)$ Th.4.2	$\Omega(n^2)$	$O(n^3)$ ✠ Th.4.5	$\Omega(n^3)$ Th.4.12
NO	YES	$O(n^2)$	$\Omega(n^2)$	$O(n^2)$ Th.4.2	$\Omega(n^2)$ Cor.4.1	$O(n^2)$ ✠ Th.4.5	$\Omega(n^2)$ Th.4.11
YES	NO	$O(n^2)$	$\Omega(n^2)$	$O(n^2)$ Th.4.3	$\Omega(n^2)$ Th.4.9	0 ✠ [FCE95b]	0 ✠ [FCE95b]
YES	YES	$O(n^2)$	$\Omega(n^2)$ Th.4.7	$O(n)$ Th.4.3	$\Omega(n)$ Th.4.10	0 ✠ [FCE95b]	0 ✠ [FCE95b]

Table 4.1: Upper and lower bounds for the number of crossings in $\langle \infty, 0, 0 \rangle$-, $\langle 0, \infty, 0 \rangle$-, and $\langle 0, 0, \infty \rangle$-drawings of clustered graphs. Flags *c-c* and *flat* mean that the clustered graph is *c-connected* and that the cluster hierarchy is *flat*, respectively. Results written in gray derive from those in black, while a "✠" means that there exist clustered graphs not admitting the corresponding drawings. A "0" occurs if the clustered graph is c-planar.

1. In Section 4.3 we provide algorithms to produce $\langle \infty, 0, 0 \rangle$-, $\langle 0, \infty, 0 \rangle$-, and $\langle 0, 0, \infty \rangle$-drawings of clustered graphs, if they exist. In particular, while $\langle \infty, 0, 0 \rangle$- and $\langle 0, \infty, 0 \rangle$-drawings always exist, we show that some clustered graphs do not admit any $\langle 0, 0, \infty \rangle$-drawing, and we present a polynomial-time algorithm to test whether a biconnected clustered graph admits a $\langle 0, 0, \infty \rangle$-drawing, which is a necessary condition for C-PLANARITY. The algorithm, whose approach is reminiscent of [ADF+12], is based on a characterization of the planar embeddings that lead to $\langle 0, 0, \infty \rangle$-drawings, and on a subsequent structural characterization of the existence of a $\langle 0, 0, \infty \rangle$-drawing for any biconnected clustered graph $C(G, T)$, based on the SPQR-tree decomposition of G.

2. The above mentioned algorithms provide upper bounds on the number of crossings for the three kinds of drawings. We show that the majority of these upper bounds are tight by providing matching lower bounds in Section 4.4. These results are summarized in Tab. 4.1.

3. In Section 4.5 we show that there are clustered graphs admitting drawings with one crossing of a certain type but requiring many crossings in drawings where only different types of crossings are allowed. For example, there are clustered graphs that admit a $\langle 1, 0, 0 \rangle$-drawing and that require $\beta \in \Omega(n^2)$ in any $\langle 0, \beta, 0 \rangle$-drawing and $\gamma \in \Omega(n^2)$ in any $\langle 0, 0, \gamma \rangle$-drawing. See Tab. 4.2 for a summary of these results.

4. In Section 4.6 we present several complexity results. Namely, we show that:

$\rightarrow$	$\langle \alpha, 0, 0 \rangle$	$\langle 0, \beta, 0 \rangle$	$\langle 0, 0, \gamma \rangle$
$\langle 1, 0, 0 \rangle$		$\Omega(n^2)$	$\Omega(n^2)$
$\langle 0, 1, 0 \rangle$	$\Omega(n)$		$\Omega(n^2)$
$\langle 0, 0, 1 \rangle$	$\Omega(n^2)$	$\Omega(n)$	

Table 4.2: Relationships between types of drawings proved in Theorem 4.14.

- minimizing $\alpha + \beta + \gamma$ in an $\langle \alpha, \beta, \gamma \rangle$-drawing is NP-complete even if the underlying graph is planar, namely a forest of star graphs;
- minimizing α in an $\langle \alpha, 0, 0 \rangle$-drawing is NP-complete even if the underlying graph is a matching;
- minimizing β in a $\langle 0, \beta, 0 \rangle$-drawing is NP-complete (see also [For05]) even for c-connected flat clustered graphs in which the underlying graph is a triconnected planar multigraph;

Section 4.2 gives definitions and preliminary lemmas, while Section 4.7 contains conclusions and open problems.

4.2 Preliminaries

We remark that every clustered graph $C(G, T)$ that is considered in this chapter is such that G is planar.

Let $C(G, T)$ be a clustered graph. If μ is an internal node of T, we denote by $V(\mu)$ the leaves of the subtree of T rooted at μ. The subgraph of G induced by $V(\mu)$ is denoted by $G(\mu)$.

Some constraints are usually enforced on the crossings among the open curves representing edges in a drawing of a graph. Namely: ($\mathcal{C}_1$) the intersections among curves form a set of isolated points; ($\mathcal{C}_2$) no three curves intersect in the same point; and ($\mathcal{C}_3$) two intersecting curves appear alternatingly in the circular order around their intersection point. Figure 4.3(a) shows a legal crossing, while Figures 4.3(b)-(d) show crossings violating Constraints $\mathcal{C}_1$, $\mathcal{C}_2$, and $\mathcal{C}_3$, respectively. These constraints naturally extend to encompass crossings involving regions representing clusters, by considering, for each region, the closed curve that forms its boundary.

Let Γ be a drawing of a clustered graph $C(G, T)$. First, we formally define the types of crossings of Γ and how to count them.

Edge-edge crossings. Each crossing between two edges of G is an *edge-edge crossing* (or *ee-crossing* for short) of Γ.

Edge-region crossings. An *edge-region crossing* (*er-crossing*) is a crossing involving an edge e of G and a region $R(\mu)$ representing a cluster μ of T; namely, if e crosses the boundary of $R(\mu)$ k times, the number of er-crossings between e and $R(\mu)$ is $\lfloor \frac{k}{2} \rfloor$. Note that, if e intersects the boundary of $R(\mu)$ exactly once, then such an intersection does not count as an er-crossing, as in the traditional C-PLANARITY literature.

Region-region crossings. A *region-region crossing* (*rr-crossing*) is a crossing involving two regions $R(\mu)$ and $R(\nu)$ representing clusters μ and ν of T, respectively, and such that μ is not an ancestor of ν and vice-versa. In fact, if μ is an ancestor of ν, then $R(\nu)$ is contained into $R(\mu)$ by the definition of drawing of a clustered graph. The number of rr-crossings between $R(\mu)$ and $R(\nu)$ is equal to the number of the topologically connected regions resulting from the relative complement of $R(\mu)$ in $R(\nu)$ (i.e., $R(\mu) \setminus R(\nu)$) minus one. Observe that, due to Constraints $\mathcal{C}_1$, $\mathcal{C}_2$, and $\mathcal{C}_3$, the number of rr-crossings between $R(\mu)$ and $R(\nu)$ is equal to the number of rr-crossings between $R(\nu)$ and $R(\mu)$. Also, as region $R(\mu)$ contains all and only the vertices of μ, intersections between regions cannot contain vertices of G. Figure 4.4 provides examples of region-region crossings.

Definition 4.1 *An $\langle \alpha, \beta, \gamma \rangle$-drawing of a clustered graph is a drawing with α ee-crossings, β er-crossings, and γ rr-crossings.*

4.3 Drawings of Clustered Graphs with Crossings

The following three sections deal with $\langle \infty, 0, 0 \rangle$-, $\langle 0, \infty, 0 \rangle$- and $\langle 0, 0, \infty \rangle$-drawings, respectively.

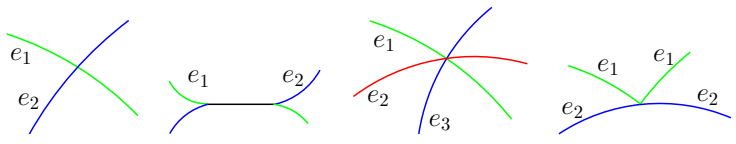


Figure 4.3: Allowed and forbidden crossings in a drawing of a graph. (a) A legal crossing. (b) A crossing violating Constraint $\mathcal{C}_1$. (c) A crossing violating Constraint $\mathcal{C}_2$. (d) A crossing violating Constraint $\mathcal{C}_3$.

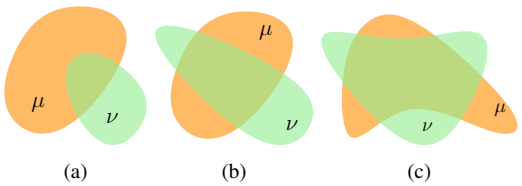


Figure 4.4: Examples of intersections between clusters generating (a) zero rr-crossings; (b) one rr-crossing; and (c) two rr-crossings.

Drawings with Edge-Edge Crossings

In this section, we show a simple algorithm to construct an $\langle \alpha, 0, 0 \rangle$-drawing of any clustered graph, in which α is asymptotically optimal in the worst case, as proved in Section 4.4.

Theorem 4.1 *Let $C(G,T)$ be a clustered graph. There exists an algorithm to compute an $\langle \alpha, 0, 0 \rangle$-drawing of $C(G,T)$ with $\alpha \in O(n^2)$.*

Proof: Let $\sigma = v_1, \ldots, v_n$ be an ordering of the vertices of G such that vertices of the same cluster are consecutive in σ. A drawing of G can be constructed as follows. Place the vertices of G along a convex curve in the order they appear in σ. Draw the edges of G as straight-line segments. Since vertices belonging to the same cluster are consecutive in σ, drawing each cluster as the convex hull of the points assigned to its vertices yields a drawing without region-region and edge-region crossings (see Fig. 4.5). Further, since G has $O(n)$ edges, and since edges are drawn as straight-line segments, such a construction produces $O(n^2)$ edge-edge crossings. □

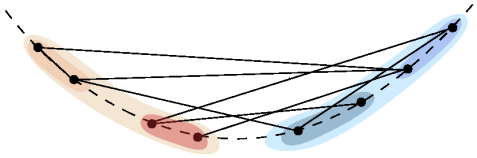


Figure 4.5: Illustration for Theorem 4.1.

Observe that, using the same construction used in the proof of Theorem 4.1, it can be proved that every clustered graph (even if its underlying graph is not planar) admits an $\langle \alpha, 0, 0 \rangle$-drawing with $\alpha \in O(n^4)$.

Drawings with Edge-Region Crossings

In this section, we show two algorithms for constructing a $\langle 0, \beta, 0\rangle$-drawing of any clustered graph $C(G,T)$, in which β is asymptotically optimal in the worst case if $C(G,T)$ is c-connected or if it is flat, as proved in Section 4.4. If $C(G,T)$ is a general clustered graph, then β is a linear factor apart from the lower bound presented in Section 4.4.

The two algorithms handle the case in which $C(G,T)$ is not c-connected (Theorem 4.2) and in which $C(G,T)$ is c-connected (Theorem 4.3), respectively. Both algorithms have three steps:

1. A spanning tree $\mathcal{T}$ of the vertices of G is constructed in such a way that, for each cluster $\mu \in T$, the subgraph of $\mathcal{T}$ induced by the vertices of μ is connected. The two algorithms construct $\mathcal{T}$ in two different ways; in particular, $\mathcal{T}$ is a subgraph of G if $C(G,T)$ is c-connected, while it is not necessarily a subgraph of G if $C(G,T)$ is not c-connected.

2. A simultaneous embedding of G and $\mathcal{T}$ is computed. A *simultaneous embedding* (*SE* for short) of two graphs $G_1(V,E_1)$ and $G_2(V,E_2)$, on the same set V of vertices, is a drawing of $G(V, E_1 \cup E_2)$ such that any crossing involves an edge from E_1 and an edge from E_2 [BCD+07].

3. A $\langle 0, \beta, 0\rangle$-drawing of $C(G,T)$ is constructed by drawing each cluster μ as a region $R(\mu)$ slightly surrounding the edges of $\mathcal{T}(\mu)$ and the regions $R(\mu_1), \ldots, R(\mu_k)$ representing the children $\mu_1, \ldots, \mu_k$ of μ.

In the case in which $C(G,T)$ is not c-connected, we get the following:

Theorem 4.2 *Let $C(G,T)$ be a clustered graph. Then, there exists an algorithm to compute a $\langle 0, \beta, 0\rangle$-drawing of $C(G,T)$ with $\beta \in O(n^3)$. If $C(G,T)$ is flat, then $\beta \in O(n^2)$.*

Proof: In the first step, the tree $\mathcal{T}$ is constructed by means of a bottom-up traversal of T. Whenever a node $\mu \in T$ is considered, a spanning tree $\mathcal{T}(\mu)$ of $V(\mu)$ is constructed as follows. Denote by $\mu_1, \ldots, \mu_k$ the children of μ in T (observe that, for each $1 \leq i \leq k$, μ_i is either a cluster or a vertex). Assume that spanning trees $\mathcal{T}(\mu_1), \ldots, \mathcal{T}(\mu_k)$ of $V(\mu_1), \ldots, V(\mu_k)$ have been already computed. The spanning tree $\mathcal{T}(\mu)$ of $V(\mu)$ is constructed by connecting a vertex of μ_1 to a vertex of each of $\mathcal{T}(\mu_2), \ldots, \mathcal{T}(\mu_k)$. Tree $\mathcal{T}$ coincides with $\mathcal{T}(\rho)$, where ρ is the root of T. Observe that some of the edges of $\mathcal{T}$ might not belong to G.

In the second step, we apply the algorithm by Kammer [Kam06] (see also [EK05]) to construct a simultaneous embedding of G and $\mathcal{T}$ in which each edge has at most two bends, which implies that each pair of edges $\langle e_1 \in G, e_2 \in \mathcal{T} \rangle$ crosses a constant number of times.

In the third step, each cluster μ is drawn as a region $R(\mu)$ slightly surrounding the edges of $\mathcal{T}(\mu)$ and the regions $R(\mu_1), \ldots, R(\mu_k)$ representing the children $\mu_1, \ldots, \mu_k$ of μ. Hence, each crossing between an edge $e_1 \in G$ and an edge $e_2 \in \mathcal{T}$ determines two intersections (hence one edge-region crossing) between e_1 and the boundary of each cluster ν such that $e_2 \in \mathcal{T}(\nu)$. Further, each edge $(u, v) \in G$ such that $(u, v) \notin \mathcal{T}$ and u and v belong to the same cluster ν, has a er-crossing with the boundary of $R(\nu)$.

Note that, for each edge $e_2 \in \mathcal{T}$, there exist $O(n)$ clusters ν such that $e_2 \in \mathcal{T}(\nu)$; also there exist $O(n^2)$ pairs of edges $\langle e_1 \in G, e_2 \in \mathcal{T} \rangle$; further, there exist $O(n)$ edges not belonging to τ; finally, for each edge $e \notin \mathcal{T}$ there exists $O(n)$ clusters ν such that both endvertices of e belong to ν. Hence, the total number of er-crossings is $O(n^3)$.

If $C(G, T)$ is flat, then for each edge $e_2 \in \mathcal{T}$ there exists at most one cluster ν different from the root such that $e_2 \in \mathcal{T}(\nu)$; also, for each edge $e \notin \mathcal{T}$ there exists at most one cluster ν different from the root such that both endvertices of e belong to ν. Hence, the total number of er-crossings is $O(n^2)$. □

If $C(G, T)$ is c-connected, we can improve the bounds of Theorem 4.2 as follows:

Theorem 4.3 *Let $C(G, T)$ be a c-connected clustered graph. Then, there exists an algorithm to compute a $\langle 0, \beta, 0 \rangle$-drawing of $C(G, T)$ with $\beta \in O(n^2)$. If $C(G, T)$ is flat, $\beta \in O(n)$.*

Proof: In the first step, the tree $\mathcal{T}$ is constructed by means of a bottom-up traversal of T. When a node $\mu \in T$ is considered, a spanning tree $\mathcal{T}(\mu)$ of $V(\mu)$ is constructed as follows. Denote by $\mu_1, \ldots, \mu_k$ the children of μ in T (note that, for each $1 \leq i \leq k$, μ_i is either a cluster or a vertex). Assume that spanning trees $\mathcal{T}(\mu_1), \ldots, \mathcal{T}(\mu_k)$ of $V(\mu_1), \ldots, V(\mu_k)$ have been already computed so that $\mathcal{T}(\mu_i)$ is a subgraph of $G(\mu_i)$, for $i = 1, \ldots, k$. Tree $\mathcal{T}(\mu)$ contains all the edges in $\mathcal{T}(\mu_1), \ldots, \mathcal{T}(\mu_k)$ plus a minimal set of edges of $G(\mu)$ connecting $\mathcal{T}(\mu_1), \ldots, \mathcal{T}(\mu_k)$. The latter set of edges always exists since $G(\mu)$ is connected. Tree $\mathcal{T}$ coincides with $\mathcal{T}(\rho)$, where ρ is the root of T. Observe that, in contrast with the construction in the proof of Theorem 4.2, all edges of $\mathcal{T}$ belong to G.

In the second step, since each edge of $\mathcal{T}$ is also an edge of G, any planar drawing of G determines a simultaneous embedding of G and $\mathcal{T}$ in which no edge of G properly crosses an edge of $\mathcal{T}$.

In the third step, clusters are drawn in the same way as in the proof of Theorem 4.2.

Note that the only edge-region crossings that may occur are those between any edge of G not in $\mathcal{T}$ whose endvertices belong to the same cluster μ and the boundary of $R(\mu)$. Since there exist $O(n)$ edges not belonging to $\mathcal{T}$ and since for each edge $e \notin \mathcal{T}$ there exist $O(n)$ clusters ν such that both endvertices of e belong to ν, it follows that the total number of edge-region crossings is $O(n^2)$.

If $C(G,T)$ is flat, then for each edge $e \notin \mathcal{T}$ there exists at most one cluster ν different from the root such that both endvertices of e belong to ν, and hence the total number of edge-region crossings is $O(n)$. □

Drawings with Region-Region Crossings

In this section, we study $\langle 0,0,\infty\rangle$-drawings of clustered graphs. First, we prove that there are clustered graphs that do not admit any $\langle 0,0,\infty\rangle$-drawings. Second, we provide a polynomial-time algorithm for testing whether a clustered graph $C(G,T)$ with G biconnected admits a $\langle 0,0,\infty\rangle$-drawing and to compute one if it exists. Third, we show an algorithm that constructs a $\langle 0,0,\gamma\rangle$-drawing Γ of any clustered graph $C(G,T)$ that admits such a drawing (the input of the algorithm is any $\langle 0,0,\infty\rangle$-drawing Γ' of $C(G,T)$) in which γ is worst-case asymptotically optimal.

To show that there exist clustered graphs not admitting any $\langle 0,0,\infty\rangle$-drawing, we give two examples. Let $C(G,T)$ be a clustered graph such that G is triconnected and has a cycle of vertices belonging to a cluster μ separating two vertices not in μ (see Fig. 4.6(a)). Note that, even in the presence of rr-crossings, one of the two vertices not in μ is enclosed by $R(\mu)$ in any $\langle 0,0,\infty\rangle$-drawing of $C(G,T)$. This example exploits the triconnectivity of the underlying graph. Next we show that even clustered graphs with series-parallel underlying graph may not admit any $\langle 0,0,\infty\rangle$-drawing. Namely, let $C(G,T)$ be a clustered graph such that G has eight vertices and is composed of parallel paths p_1, p_2, p_3, and p_4. Tree T is such that cluster μ_1 contains a vertex of p_1 and a vertex of p_2; cluster μ_2 contains a vertex of p_2 and a vertex of p_3; cluster μ_3 contains a vertex of p_2 and a vertex of p_4 (see Fig. 4.6(b)). Note that, in any $\langle 0,0,\infty\rangle$-drawing of $C(G,T)$, path p_2 should be adjacent to all the other paths in the order around the poles, and this is not possible by the planarity of the drawing of G.

Since some clustered graphs do not admit any $\langle 0,0,\infty\rangle$-drawing, we study the complexity of testing whether a clustered graph $C(G,T)$ admits one. In order to do that, we first give a characterization of the planar embeddings of G that allow for the realization of a $\langle 0,0,\infty\rangle$-drawing of $C(G,T)$. Namely, let $C(G,T)$ be a clustered graph and let Γ be a planar embedding of G. For each cluster $\mu \in T$ consider an auxiliary graph $H(\mu)$ with the same vertices as $G(\mu)$ and such that there

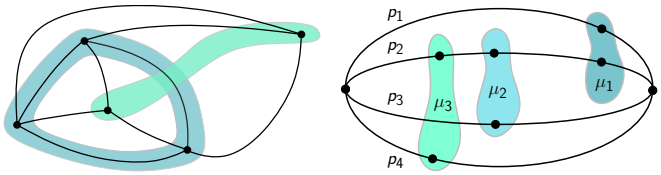


Figure 4.6: Two clustered graphs not admitting any $\langle 0, 0, \infty \rangle$-drawing. The underlying graph of (a) is a triconnected planar graph, while the underlying graph of (b) is a series-parallel graph.

is an edge between two vertices of $H(\mu)$ if and only if the corresponding vertices of G are incident to the same face in Γ.

Lemma 4.1 *Let $C(G,T)$ be a clustered graph and let Γ be a planar embedding of G. Then, $C(G,T)$ admits a $\langle 0, 0, \infty \rangle$-drawing preserving Γ if and only if, for each cluster $\mu \in T$: (i) graph $H(\mu)$ is connected and (ii) there exists no cycle of G whose vertices belong to μ and whose interior contains in Γ a vertex not belonging to μ.*

Proof: We first prove the necessity of the conditions. For the necessity of Condition (i), suppose that $H(\mu)$ is not connected. Then, for any two distinct connected components $H_1(\mu)$ and $H_2(\mu)$ of $H(\mu)$, there exists a cycle $\mathcal{C}$ in G separating $H_1(\mu)$ and $H_2(\mu)$, as otherwise $H_1(\mu)$ and $H_2(\mu)$ would be incident to a common face, hence they would not be distinct connected components of $H(\mu)$. Therefore, the boundary of any region $R(\mu)$ representing μ intersects (at least) one of the edges of $\mathcal{C}$. For the necessity of Condition (ii), suppose that a cycle $\mathcal{C}$ exists in Γ whose vertices belong to μ and whose interior contains in Γ a vertex not belonging to μ. Then, in any drawing of $R(\mu)$ as a simple closed region containing all and only the vertices in μ, the border of $R(\mu)$ intersects (at least) one edge of $\mathcal{C}$.

We next prove the sufficiency of the conditions. Suppose that Conditions (i) and (ii) hold. Consider any subgraph $H'(\mu)$ of $H(\mu)$ such that: (a) $G(\mu) \subseteq H'(\mu)$; (b) $H'(\mu)$ is connected; and (c) for every cycle $\mathcal{C}$ in $H'(\mu)$, if any, all the edges of $\mathcal{C}$ belong to G. Observe that the fact that $H(\mu)$ satisfies conditions (i) and (ii) implies the existence of a graph $H'(\mu)$ satisfying (a), (b), and (c). Draw each edge of $H'(\mu)$ not in G inside the corresponding face. Represent μ as a region slightly surrounding the (possibly non-simple) cycle delimiting the outer face of $H'(\mu)$. Denote by Γ'_C the resulting drawing and denote by Γ_C the drawing of $C(G,T)$ obtained from Γ'_C by

removing the edges not in G. We have that Γ_C contains no ee-crossing, since Γ is a planar embedding. Also, it contains no er-crossing, since the only edges crossing clusters in Γ'_C are those belonging to $H'(\mu)$ and not belonging to $G(\mu)$. □

Our next goal is to provide an algorithm that, given a clustered graph $C(G,T)$ such that G is biconnected, tests whether G admits a planar embedding allowing for a $\langle 0,0,\infty\rangle$-drawing of $C(G,T)$.

We start by giving some definitions. In the reminder of the chapter, even when not explicitly mentioned, we will always assume the considered SPQR-trees to be rooted at an edge of the graph. Let $C(G,T)$ be a clustered graph such that G is biconnected and consider the SPQR-tree $\mathcal{T}$ of G rooted at any Q-node ρ. The choice of rooting $\mathcal{T}$ at ρ corresponds to only consider planar embeddings of G in which the edge e_ρ of G corresponding to ρ is incident to the outer face.

Consider a node $\tau \in \mathcal{T}$, its pertinent graph $pert(\tau)$ (augmented with an edge e between the poles of τ, representing the parent of τ), and a planar embedding $\Gamma(pert(\tau))$ with e on the outer face. Observe that assuming e to be incident to the outer face of $\Gamma(pert(\tau))$ is not a loss of generality, given that e_ρ is assumed to be incident to the outer face of every considered planar embedding of G. Namely, consider any planar embedding Γ_G of G in which e_ρ is incident to the outer face and let $\Gamma^-(pert(\tau))$ be the embedding of $pert(\tau)$ (except for edge e) obtained by restricting Γ_G to $pert(\tau)$. Then, the subgraph of G not in $pert(\tau)$ (i.e., the "rest of the graph" with respect to τ) lies in Γ_G in the outer face of $\Gamma^-(pert(\tau))$, hence edge e can be inserted in the outer face of $\Gamma^-(pert(\tau))$, thus obtaining a planar embedding $\Gamma(pert(\tau))$ with e on the outer face.

Let $f'(\tau)$ and $f''(\tau)$ be the two faces of $\Gamma(pert(\tau))$ that are incident to e. For each cluster $\mu \in T$, we define an auxiliary graph $H(\tau,\mu)$ as the graph containing all the vertices of $pert(\tau)$ that belong to μ and such that two vertices of $H(\tau,\mu)$ are connected by an edge if and only if they are incident to the same face in $\Gamma(pert(\tau))$. Observe that $H(\rho,\mu)$ coincides with the above defined auxiliary graph $H(\mu)$. Also, observe that no two connected components of $H(\tau,\mu)$ exist both containing a vertex incident to $f'(\tau)$ or both containing a vertex incident to $f''(\tau)$.

Next we introduce some classifications of the nodes of $\mathcal{T}$ and of the embeddings of their pertinent graphs that will be used to find an embedding of G such that Conditions (i) and (ii) of Lemma 4.1 are satisfied for each cluster μ.

In order to keep track of the connectivity of $H(\mu)$ (Condition (i) of Lemma 4.1), for each node $\tau \in \mathcal{T}$ and for each cluster $\mu \in T$, we say that $\Gamma(pert(\tau))$ is:

***μ-traversable*:** if $H(\tau,\mu)$ is connected and contains at least one vertex incident to $f'(\tau)$ and one vertex incident to $f''(\tau)$ (see Fig. 4.7(a)).

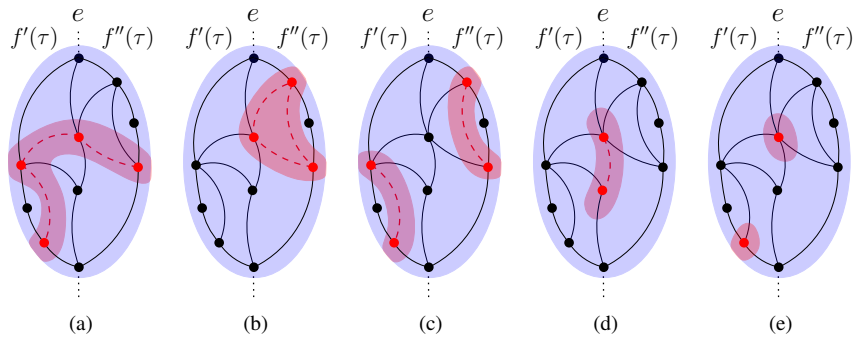


Figure 4.7: Examples of embeddings of $\Gamma(pert(\tau))$: (a) μ-traversable; (b) μ-sided; (c) μ-bisided; (d) μ-kernelized; (e) μ-infeasible. Dashed red edges belong to $H(\tau, \mu)$. The five drawings represent five embeddings $\Gamma(pert(\tau))$ for five different graphs $pert(\tau)$.

***μ-sided*:** if $H(\tau, \mu)$ is connected and contains at least one vertex incident to $f'(\tau)$ and no vertex incident to $f''(\tau)$, or vice versa (see Fig. 4.7(b)).

***μ-bisided*:** if $H(\tau, \mu)$ consists of two connected components, one containing a vertex of $f'(\tau)$ and the other one containing a vertex of $f''(\tau)$ (see Fig. 4.7(c)).

***μ-kernelized*:** if $H(\tau, \mu)$ is connected and contains neither a vertex incident to $f'(\tau)$ nor a vertex incident to $f''(\tau)$ (see Fig. 4.7(d)).

***μ-infeasible*:** if $H(\tau, \mu)$ has at least two connected components of which one has no vertex incident to $f'(\tau)$ or $f''(\tau)$ (see Fig. 4.7(e)).

Note that, if τ contains at least one vertex of μ, then $\Gamma(pert(\tau))$ is exactly of one of the types of embedding defined above.

First, we derive an elementary condition that prevents an embedding from being μ-traversable.

Lemma 4.2 *Suppose that an embedding $\Gamma(pert(\tau))$ of $pert(\tau)$ with e incident to the outer face is neither μ-traversable nor μ-infeasible. Then, there exists a path in $pert(\tau)$ that connects the poles of τ, that is different from e, and none of whose vertices belongs to μ.*

Proof: Suppose first that $H(\tau, \mu)$ is connected in $\Gamma(pert(\tau))$. Then, $\Gamma(pert(\tau))$ is either μ-traversable, or μ-kernelized, or μ-sided, by definition. By assumption,

$\Gamma(pert(\tau))$ is not μ-traversable, hence it is either μ-kernelized or μ-sided. In both cases, the path of $\Gamma(pert(\tau))$ delimiting $f'(\tau)$, different from e, and connecting the poles of τ, or the path of $\Gamma(pert(\tau))$ delimiting $f''(\tau)$, different from e, and connecting the poles of τ is such that none of its vertices belongs to μ, and the statement follows.

Suppose next that $H(\tau, \mu)$ is not connected in $\Gamma(pert(\tau))$. Then, $\Gamma(pert(\tau))$ is either μ-infeasible or μ-bisided, by definition. By assumption, $\Gamma(pert(\tau))$ is not μ-infeasible, hence it is μ-bisided. By definition of $H(\tau, \mu)$, the two connected components $H'(\tau, \mu)$ and $H''(\tau, \mu)$ of $H(\tau, \mu)$ are not incident to any common face. Thus, there exists a cycle $\mathcal{C}$ in $\Gamma(pert(\tau))$ containing such components on different sides and none of whose vertices belongs to μ. Cycle $\mathcal{C}$ contains edge e, given that $H'(\tau, \mu)$ and $H''(\tau, \mu)$ contain vertices incident to $f'(\tau)$ and $f''(\tau)$. It follows that the path obtained from $\mathcal{C}$ by removing edge e connects the poles of τ, is different from edge e, and is such that none of its vertices belongs to μ, thus proving the statement. □

While the fact that $\Gamma(pert(\tau))$ is μ-sided, μ-bisided, μ-kernelized, or μ-infeasible does not rule the possibility that a different planar embedding of $pert(\tau)$ is of a different type, if $\Gamma(pert(\tau))$ is μ-traversable then any other embedding of $pert(\tau)$ is either μ-traversable or μ-infeasible, as proved in the following.

Lemma 4.3 *Let $\Gamma_1(pert(\tau))$ and $\Gamma_2(pert(\tau))$ be two planar embeddings of $pert(\tau)$, both having edge e incident to the outer face. If $\Gamma_1(pert(\tau))$ is μ-traversable, then $\Gamma_2(pert(\tau))$ is either μ-traversable or μ-infeasible.*

Proof: Suppose, for a contradiction, that $\Gamma_2(pert(\tau))$ is neither μ-traversable nor μ-infeasible. By Lemma 4.2, there exists a path in $pert(\tau)$ that connects the poles of τ, that is different from e, and none of whose vertices belongs to μ. It follows that the auxiliary graph $H(\tau, \mu)$ associated with $\Gamma_2(pert(\tau))$ is disconnected, or does not contain a vertex incident to $f'(\tau)$, or does not contain a vertex incident to $f''(\tau)$. □

By Lemma 4.3, if an embedding of $pert(\tau)$ is μ-traversable, then every embedding of $pert(\tau)$ which is not μ-infeasible is μ-traversable. Hence, if a μ-traversable embedding of $pert(\tau)$ exists, we say that τ and the virtual edge representing τ in the skeleton of the parent of τ in $\mathcal{T}$ is μ-traversable.

Next, we introduce some definitions used to deal with Condition (ii) of Lemma 4.1. Namely, for each node $\tau \in \mathcal{T}$ and for each cluster $\mu \in T$, we say that τ (and the virtual edge representing τ in the skeleton of its parent) is:

***μ-touched*:** if there exists a vertex in $pert(\tau) \setminus \{u, v\}$ that belongs to μ.

***μ-full*:** if all the vertices in $pert(\tau)$ belong to μ.

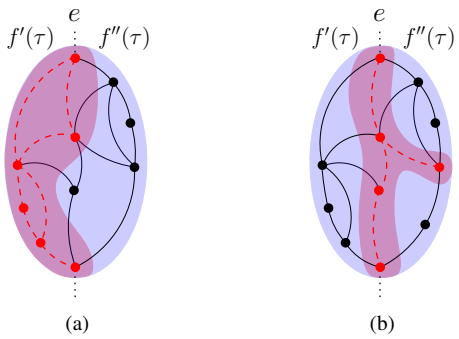


Figure 4.8: Examples of embeddings of $\Gamma(pert(\tau))$: (a) μ-side-spined; (b) μ-central-spined. Dashed red edges belong to $H(\tau, \mu)$. The two drawings represent two embeddings $\Gamma(pert(\tau))$ for two different graphs $pert(\tau)$.

***μ-spined*:** if there exists in $pert(\tau)$ a path P between the poles of τ different from e and containing only vertices of μ. Observe that, if τ is μ-spined and $pert(\tau)$ is not a single edge, then τ is μ-touched.

Given a μ-spined node τ, an embedding $\Gamma(pert(\tau))$ is **μ-side-spined** if at least one of the two paths different from edge e, connecting the poles of τ, and delimiting the outer face of $\Gamma(pert(\tau))$ has only vertices in μ (see Fig 4.8(a)). Otherwise, it is **μ-central-spined** (see Fig 4.8(b)).

Observe that, if τ is μ-spined, then it is also μ-traversable, since its poles belong to μ.

We say that an embedding $\Gamma(pert(\tau))$ of $pert(\tau)$ with e incident to the outer face is *extensible* if the following condition holds: If $C(G,T)$ admits a $\langle 0,0,\infty\rangle$-drawing in which the edge e_ρ corresponding to ρ is incident to the outer face, then it admits a $\langle 0,0,\infty\rangle$-drawing in which e_ρ is incident to the outer face and in which the embedding of $pert(\tau)$ is $\Gamma(pert(\tau))$. Observe that, if an embedding $\Gamma(pert(\tau))$ of $pert(\tau)$ is μ-infeasible, for some $\mu \in T$, then $\Gamma(pert(\tau))$ is not extensible.

One key ingredient of our result is that if an embedding $\Gamma(pert(\tau))$ of $pert(\tau)$ is extensible, not only there exists a $\langle 0,0,\infty\rangle$-drawing of $C(G,T)$ in which the embedding of $pert(\tau)$ is $\Gamma(pert(\tau))$, but for *every* $\langle 0,0,\infty\rangle$-drawing Γ of $C(G,T)$ in which e_ρ is incident to the outer face, the embedding of $pert(\tau)$ in Γ can be modified to be $\Gamma(pert(\tau))$, without changing the rest of Γ, while maintaining the property that Γ is a $\langle 0,0,\infty\rangle$-drawing. In the following we formalize such a claim.

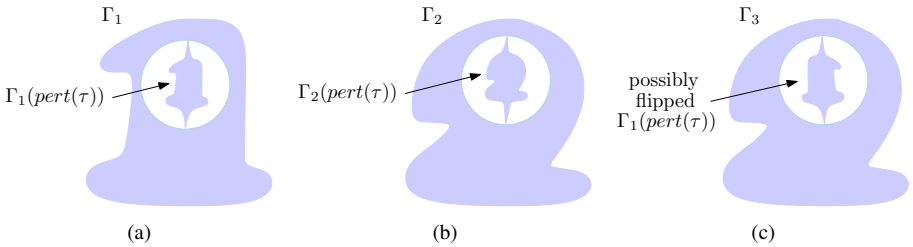


Figure 4.9: Illustration for the statement of Lemma 4.4. (a) Embedding Γ_1. (b) Embedding Γ_2. (c) Embedding Γ_3.

Lemma 4.4 *Let $C(G,T)$ be a clustered graph, with G biconnected, that admits a $\langle 0,0,\infty\rangle$-drawing in which the edge e_ρ corresponding to the root ρ of SPQR-tree $\mathcal{T}$ of G is incident to the outer face. Let Γ_1 and Γ_2 be two $\langle 0,0,\infty\rangle$-drawings of $C(G,T)$ in which e_ρ is incident to the outer face. Let τ be a P-node or an R-node of $\mathcal{T}$. Let $\Gamma_i(pert(\tau))$ be the embedding of $pert(\tau)$ minus edge e in Γ_i, for $i=1,2$. Let Γ_3 be the drawing obtained from Γ_2 by replacing $\Gamma_2(pert(\tau))$ with $\Gamma_1(pert(\tau))$, possibly after performing a flip of $\Gamma_1(pert(\tau))$ (see Fig. 4.9(c)). Then, Γ_3 is a $\langle 0,0,\infty\rangle$-drawing of $C(G,T)$ in which e_ρ is incident to the outer face.*

In the following we prove Lemma 4.4. Namely, we prove that, after the replacement of $\Gamma_2(pert(\tau))$ with $\Gamma_1(pert(\tau))$ and, possibly, a flip of $\Gamma_1(pert(\tau))$, the resulting embedding Γ_3 of G satisfies the conditions of Lemma 4.1. Observe that e_ρ is incident to the outer face of Γ_3, given that it is incident to the outer face of Γ_2.

We introduce some terminology. The *rest of the graph with respect to τ* is the graph $G(\overline{\tau})$ obtained from G by removing the vertices of $pert(\tau)$ different from its poles and by inserting a dummy edge e_τ between the poles of $pert(\tau)$. In terms of SPQR-trees, the rest of the graph can be equivalently defined as follows. Denote by τ' the parent of τ in $\mathcal{T}$. Then, the rest of the graph is the pertinent graph of τ' in any re-rooting of $\mathcal{T}$ in which τ' becomes a child of τ plus an edge e_τ between the poles of τ. Denote by $f'(\tau)$ and $f''(\tau)$ the faces of $G(\overline{\tau})$ incident to e_τ. Recall that $f'(\tau)$ and $f''(\tau)$ also denote the faces incident to e in the embeddings $\Gamma_1(pert(\tau))$ and $\Gamma_2(pert(\tau))$ of $pert(\tau)$ in which edge e is added in the outer face. We further overload the notation $f'(\tau)$ and $f''(\tau)$ to let them represent the faces of Γ_i that are shared by $\Gamma_i(pert(\tau))$ and $\Gamma_i(G(\overline{\tau}))$, for $i=1,2,3$. For any node μ of T, the *auxiliary graph $H(\overline{\tau},\mu)$ of $G(\overline{\tau})$* in Γ_i, for any $i=1,2,3$, is the graph containing all the vertices of $G(\overline{\tau})$ that belong to μ and such that two vertices of $H(\overline{\tau},\mu)$ are connected by an edge

if and only if they are incident to the same face in Γ_i. Let $\Gamma_i(G(\overline{\tau}))$ be the embedding of $G(\overline{\tau})$ in Γ_i, for $i = 1, 2, 3$.

The definitions of μ-traversable, μ-sided, μ-bisided, μ-kernelized, μ-infeasible, μ-touched, μ-full, and μ-spined apply to the rest of the graph with respect to τ analogously as to the pertinent graph of τ. For example, $G(\overline{\tau})$ is *μ-traversable* in Γ_i if $H(\overline{\tau}, \mu)$ is connected and contains at least one vertex incident to $f'(\tau)$ and one vertex incident to $f''(\tau)$.

For any $i = 1, 2$, if Γ_i is such that $\Gamma_i(pert(\tau))$ is μ-sided or μ-side-spined, then, we denote by $p(\Gamma_i, \tau, \mu)$ the path that (i) connects the poles of τ, (ii) belongs to $pert(\tau)$, (iii) delimits $f'(\tau)$ or $f''(\tau)$ in Γ_i, and (iv) contains vertices of μ (if $\Gamma_i(pert(\tau))$ is μ-sided, see Fig. 4.10(a)), or entirely belongs to μ (if $\Gamma_i(pert(\tau))$ is μ-side-spined, see Fig. 4.10(b)).

Similarly, for any $i = 1, 2$, suppose Γ_i is such that $\Gamma_i(G(\overline{\tau}))$ is μ-sided, or is μ-side-spined, or is μ-full and both the poles of τ belong to the outer face of Γ_i (refer to Figs. 4.10(c), 4.10(d), and 4.10(e) for examples of the three cases, respectively). Then, we denote by $p(\Gamma_i, \overline{\tau}, \mu)$ the path that (i) connects the poles of τ, (ii) belongs to $G(\overline{\tau})$, (iii) delimits $f'(\tau)$ or $f''(\tau)$ in Γ_i, and (iv) contains vertices of μ (if $\Gamma_i(G(\overline{\tau}))$ is μ-sided, see Fig. 4.10(c)), or entirely belongs to μ (if $\Gamma_i(G(\overline{\tau}))$ is μ-side-spined, see Fig. 4.10(d)), or is not entirely incident to the outer face of Γ_i (if $\Gamma_i(G(\overline{\tau}))$ is μ-full and both the poles of τ belong to the outer face of Γ_i, see Fig. 4.10(e)).

We show a simple algorithm to determine a flip of $\Gamma_1(pert(\tau))$; observe that the choice of such a flip completely determines Γ_3, as the embedding of $G(\overline{\tau})$ in Γ_3 coincides with $\Gamma_2(G(\overline{\tau}))$. Consider any cluster $\mu \in T$ such that one of the following holds:

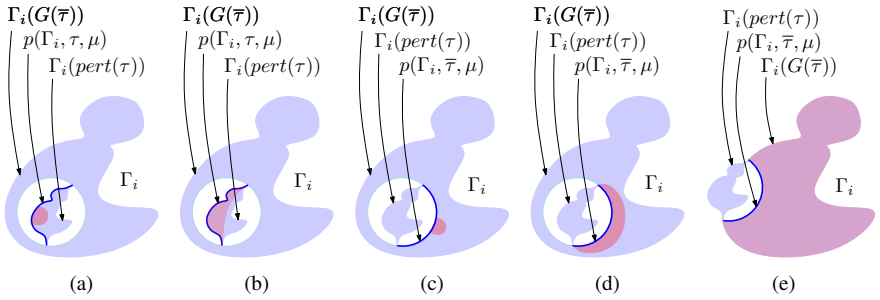


Figure 4.10: Illustration for the definition of paths $p(\Gamma_i, \tau, \mu)$ and $p(\Gamma_i, \overline{\tau}, \mu)$ of Γ_i, with $i = 1, 2$.

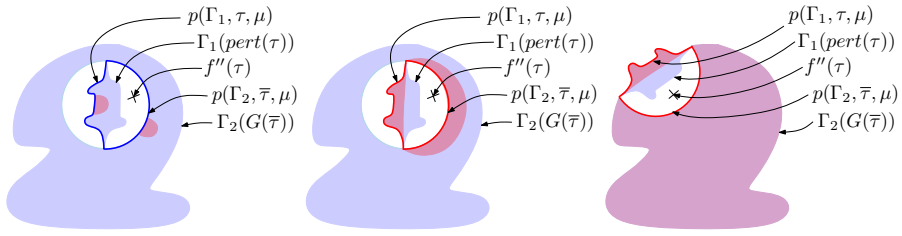


Figure 4.11: The three cases of the proof of Lemma 4.4. (a) Case 1. (b) Case 2. (c) Case 3.

Case 1: $\Gamma_1(pert(\tau))$ and $\Gamma_2(G(\overline{\tau}))$ are both μ-sided (see Fig. 4.11(a)).

Case 2: $\Gamma_1(pert(\tau))$ and $\Gamma_2(G(\overline{\tau}))$ are both μ-side-spined and not μ-full (see Fig. 4.11(b)).

Case 3: $G(\overline{\tau})$ is μ-full, $\Gamma_1(pert(\tau))$ is μ-side-spined and not μ-full, and both the poles of τ belong to the outer face of Γ_2 (see Fig. 4.11(c)).

Then, flip $\Gamma_1(pert(\tau))$ so that $p(\Gamma_1,\tau,\mu)$ and $p(\Gamma_2,\overline{\tau},\mu)$ delimit a face in Γ_3.

If no cluster $\mu \in T$ exists that determines the flip of $\Gamma_1(pert(\tau))$, that is, if for all clusters of T none of the above cases applies, then arbitrarily choose a flip for $\Gamma_1(pert(\tau))$. We will prove later that no two clusters $\mu \neq \nu \in T$ exist that determine different flips for $\Gamma_1(pert(\tau))$.

We now proceed with the proof of Lemma 4.4, showing that the embedding Γ_3 of G satisfies the conditions of Lemma 4.1.

Claim 4.1 *Embedding Γ_3 satisfies Condition (i) of Lemma 4.1.*

Proof: Assume, for a contradiction, that Condition (i) of Lemma 4.1 is not satisfied by Γ_3. Consider any node $\mu \in T$ such that $H(\mu)$ is not connected in Γ_3.

If $pert(\tau)$ or $G(\overline{\tau})$ is not μ-touched, then the connectivity of $H(\mu)$ in Γ_3 descends from the connectivity of $H(\mu)$ in Γ_2 or in Γ_1, respectively, a contradiction.

If $\Gamma_1(pert(\tau))$ is μ-infeasible, then Γ_1 is not a $\langle 0,0,\infty\rangle$-drawing, a contradiction. Analogously, if $\Gamma_2(G(\overline{\tau}))$ is μ-infeasible, then Γ_2 is not a $\langle 0,0,\infty\rangle$-drawing, a contradiction.

If $\Gamma_1(pert(\tau))$ is μ-kernelized, then, since $H(\mu)$ is connected in Γ_1, it follows that $G(\overline{\tau})$ is not μ-touched, a case that has been already addressed. Analogous considerations apply if $\Gamma_2(G(\overline{\tau}))$ is μ-kernelized.

If $\Gamma_1(pert(\tau))$ is μ-traversable, then by Lemma 4.3 we have that $\Gamma_2(pert(\tau))$ is either μ-infeasible or μ-traversable. In the former case, we have that Γ_2 is not a $\langle 0, 0, \infty \rangle$-drawing, a contradiction. In the latter case, if $H(\mu)$ is not connected in Γ_3, then $\Gamma_2(G(\overline{\tau}))$ is μ-kernelized or μ-infeasible, hence $H(\mu)$ is not connected in Γ_2, a contradiction. Analogous considerations apply if $\Gamma_2(G(\overline{\tau}))$ is μ-traversable.

If $\Gamma_1(pert(\tau))$ is μ-bisided, then $\Gamma_1(G(\overline{\tau}))$ is μ-traversable. Then, by Lemma 4.3 we have that either $\Gamma_2(G(\overline{\tau}))$ is either μ-infeasible or μ-traversable. In both cases, we have already shown how to derive a contradiction. Analogous considerations apply if $\Gamma_2(G(\overline{\tau}))$ is μ-bisided.

Hence, the only case that remains to be considered is the one in which both $\Gamma_1(pert(\tau))$ and $\Gamma_2(G(\overline{\tau}))$ are μ-sided. Observe that Case 1 for the determination of the flip of $\Gamma_1(pert(\tau))$ applies, hence $\Gamma_1(pert(\tau))$ is flipped in such a way that the connected component of $H(\mu)$ induced by the vertices in $\Gamma_1(pert(\tau))$ and the connected component of $H(\mu)$ induced by the vertices in $\Gamma_2(G(\overline{\tau}))$ both contain vertices incident to either $f'(\tau)$ or $f''(\tau)$, hence they are connected in Γ_3. It follows that $H(\mu)$ is connected in Γ_3, a contradiction. □

Claim 4.2 *Embedding Γ_3 satisfies Condition (ii) of Lemma 4.1.*

Proof: Assume for a contradiction that Condition (ii) of Lemma 4.1 is not satisfied by Γ_3. Then, for some cluster $\mu \in T$, we have that Γ_3 contains a cycle $\mathcal{C}$ whose vertices belong to μ and whose interior contains in Γ_3 a vertex not belonging to μ.

Suppose first that all the edges of $\mathcal{C}$ belong to $G(\overline{\tau})$. Since the embedding $\Gamma_3(G(\overline{\tau}))$ of $G(\overline{\tau})$ in Γ_3 coincides with the embedding $\Gamma_2(G(\overline{\tau}))$ of $G(\overline{\tau})$ in Γ_2, it follows that $\mathcal{C}$ contains a vertex not belonging to μ in its interior in Γ_2. This contradicts Lemma 4.1.

Suppose next that all the edges of $\mathcal{C}$ belong to $pert(\tau)$. Since the embedding $\Gamma_3(pert(\tau))$ of $pert(\tau)$ in Γ_3 coincides with the embedding $\Gamma_1(pert(\tau))$ of $pert(\tau)$ in Γ_1, up to a flip, it follows that $\mathcal{C}$ contains a vertex not belonging to μ in its interior in Γ_1. This contradicts Lemma 4.1.

We can hence assume that $\mathcal{C}$ contains both edges of $pert(\tau)$ and edges of $G(\overline{\tau})$. That is, $\mathcal{C}$ is composed of a path $q(\tau, \mu)$ in $pert(\tau)$ connecting the poles of τ and of a path $q(\overline{\tau}, \mu)$ in $G(\overline{\tau}) - e_\tau$ connecting the poles of τ. Thus, each of $\Gamma_1(pert(\tau))$ and $\Gamma_2(G(\overline{\tau}))$ is either μ-full, or μ-central-spined, or μ-side-spined and not μ-full.

If $\Gamma_1(pert(\tau))$ is μ-central-spined, then there exist two vertices x and y that belong to $pert(\tau)$, that do not belong to μ, and that lie on different sides of $\mathcal{C}$ in Γ_1. This contradicts Lemma 4.1. Analogously, if $\Gamma_2(G(\overline{\tau}))$ is μ-central-spined, then there exist two vertices x and y that belong to $G(\overline{\tau})$, that do not belong to μ, and that lie on different sides of $\mathcal{C}$ in Γ_2. This contradicts Lemma 4.1.

If $G(\overline{\tau})$ is μ-full and $pert(\tau)$ is μ-full, then $\mathcal{C}$ does not contain any vertex not belonging to μ in its interior, a contradiction.

If $pert(\tau)$ is μ-full and $\Gamma_2(G(\overline{\tau}))$ is μ-side-spined and not μ-full, then any vertex not belonging to μ belongs to $G(\overline{\tau})$. Since the embedding $\Gamma_3(G(\overline{\tau}))$ of $G(\overline{\tau})$ in Γ_3 coincides with the embedding $\Gamma_2(G(\overline{\tau}))$ of $G(\overline{\tau})$ in Γ_2, it follows that $\mathcal{C}$ contains a vertex not belonging to μ in its interior in Γ_2. This contradicts Lemma 4.1.

If $\Gamma_1(pert(\tau))$ and $\Gamma_2(G(\overline{\tau}))$ are both μ-side-spined and not μ-full, or if $G(\overline{\tau})$ is μ-full, $\Gamma_1(pert(\tau))$ is μ-side-spined and not μ-full, and both the poles of τ belong to the outer face of Γ_2, then Case 2 or Case 3 for the determination of the flip of $\Gamma_1(pert(\tau))$ applies, respectively, hence $\Gamma_1(pert(\tau))$ is flipped in such a way that $p(\Gamma_1, \tau, \mu)$ and $p(\Gamma_2, \overline{\tau}, \mu)$ are both incident to either $f'(\tau)$ or $f''(\tau)$ in Γ_3. Since Γ_1 is a $\langle 0, 0, \infty \rangle$-drawing of $C(G, T)$, by Lemma 4.1 the cycle composed of paths $p(\Gamma_1, \tau, \mu)$ and $q(\tau, \mu)$ does not contain any vertex not belonging to μ in its interior. Hence, $\mathcal{C}$ contains a vertex not belonging to μ in its interior if and only if the cycle $\mathcal{C}'$ composed of $q(\overline{\tau}, \mu)$ and $p(\Gamma_1, \tau, \mu)$ contains a vertex not belonging to μ in its interior. Also, since $p(\Gamma_1, \tau, \mu)$ and $p(\Gamma_2, \overline{\tau}, \mu)$ both delimit a face of Γ_3, we have that $\mathcal{C}'$ contains a vertex not belonging to μ in its interior if and only if the cycle $\mathcal{C}''$ composed of $q(\overline{\tau}, \mu)$ and $p(\Gamma_2, \overline{\tau}, \mu)$ contains a vertex not belonging to μ in its interior in Γ_3. Observe that $\mathcal{C}''$ is a cycle in $G(\overline{\tau})$. Hence, if $\mathcal{C}''$ contains a vertex not belonging to μ in its interior in Γ_3, then it contains a vertex not belonging to μ in its interior in Γ_2, given that the embedding $\Gamma_3(G(\overline{\tau}))$ of $G(\overline{\tau})$ in Γ_3 coincides with the embedding $\Gamma_2(G(\overline{\tau}))$ of $G(\overline{\tau})$ in Γ_2. This contradicts Lemma 4.1.

Finally, if $G(\overline{\tau})$ is μ-full, $pert(\tau)$ is not μ-full, and at least one of the poles of τ does not belong to the outer face of Γ_2, then there exists a cycle $\mathcal{C}'$ in $G(\overline{\tau})$ that contains all the vertices of $pert(\tau)$, except possibly for its poles, in its interior. All the vertices of $\mathcal{C}'$ belong to μ; further, at least one vertex of $pert(\tau)$ does not belong to μ. This contradicts Lemma 4.1. □

In order to conclude the proof of Lemma 4.4, it remains to prove that no two clusters $\mu \neq \nu \in T$ exist that determine different flips for $\Gamma_1(pert(\tau))$.

Claim 4.3 *No two clusters $\mu \neq \nu \in T$ exist that determine different flips for $\Gamma_1(pert(\tau))$ when constructing Γ_3.*

Proof: Assume, for a contradiction, that two distinct clusters μ and ν determine different flips for $\Gamma_1(pert(\tau))$.

- Suppose that Case 2 or Case 3 applies to μ and that Case 2 or 3 applies to ν to determine a different flip for $\Gamma_1(pert(\tau))$. Since the poles of τ are both

contained in μ and in ν, it follows that μ is an ancestor of ν or ν is an ancestor of μ. Assume the former, the discussion for the latter case being analogous.

- Assume first that $p(\Gamma_1, \tau, \mu)$ and $p(\Gamma_1, \tau, \nu)$ are distinct. Since μ is an ancestor of ν, either all vertices of $pert(\tau)$ belong to μ, hence τ is μ-full, thus contradicting the fact that Case 2 or Case 3 applies to μ, or the cycle composed of $p(\Gamma_1, \tau, \mu)$ and $p(\Gamma_1, \tau, \nu)$ entirely belongs to μ and contains in its interior a vertex not belonging to μ. This contradicts Lemma 4.1.
- Assume next that $p(\Gamma_1, \tau, \mu)$ and $p(\Gamma_1, \tau, \nu)$ are the same path.
  - If Case 2 applies to μ, then Case 2 applies to ν as well. In fact, $G(\overline{\tau})$ is not ν-full, given that μ is an ancestor of ν and given that $G(\overline{\tau})$ is not μ-full. If $p(\Gamma_2, \overline{\tau}, \mu)$ and $p(\Gamma_2, \overline{\tau}, \nu)$ are distinct, then either all vertices of $pert(\tau)$ belong to μ, hence τ is μ-full, thus contradicting the fact that Case 2 applies to μ, or the cycle composed of $p(\Gamma_2, \overline{\tau}, \mu)$ and $p(\Gamma_2, \overline{\tau}, \nu)$ entirely belongs to μ and contains in its interior a vertex not belonging to μ. This contradicts Lemma 4.1. Hence, $p(\Gamma_2, \overline{\tau}, \mu)$ and $p(\Gamma_2, \overline{\tau}, \nu)$ are the same path. However, since $p(\Gamma_1, \tau, \mu) = p(\Gamma_1, \tau, \nu)$ and $p(\Gamma_2, \overline{\tau}, \mu) = p(\Gamma_2, \overline{\tau}, \nu)$, clusters μ and ν determine the same flip for $\Gamma_1(pert(\tau))$, a contradiction to the assumptions.
  - If Case 3 applies to μ and Case 2 applies to ν, then we argue as follows. If $p(\Gamma_2, \overline{\tau}, \mu)$ and $p(\Gamma_2, \overline{\tau}, \nu)$ are distinct, then $p(\Gamma_2, \overline{\tau}, \nu)$ delimits the outer face of Γ_2, given that Case 3 applies to μ and given that $p(\Gamma_2, \overline{\tau}, \mu)$ does not delimit the outer face of Γ_2. Thus, the cycle $\mathcal{C}$ composed of $p(\Gamma_2, \overline{\tau}, \nu)$ and of any path in $pert(\tau)$ connecting the poles of τ and entirely belonging to ν (such a path exists given that $\Gamma_1(pert(\tau))$ is ν-spined) contains in its interior in Γ_2 all the vertices of $G(\overline{\tau})$ not in $p(\Gamma_2, \overline{\tau}, \nu)$. Since $G(\overline{\tau})$ is not ν-full, $\mathcal{C}$ entirely belongs to ν and contains in its interior a vertex not belonging to ν. This contradicts Lemma 4.1. If $p(\Gamma_2, \overline{\tau}, \mu)$ and $p(\Gamma_2, \overline{\tau}, \nu)$ are the same path, then, since $p(\Gamma_1, \tau, \mu) = p(\Gamma_1, \tau, \nu)$ and $p(\Gamma_2, \overline{\tau}, \mu) = p(\Gamma_2, \overline{\tau}, \nu)$, clusters μ and ν determine the same flip for $\Gamma_1(pert(\tau))$, a contradiction to the assumptions.
  - If Case 3 applies to both μ and ν, then $p(\Gamma_2, \overline{\tau}, \mu) = p(\Gamma_2, \overline{\tau}, \nu)$ is the path that connects the poles of τ, belongs to $G(\overline{\tau})$, and does not delimit the outer face of Γ_2. Since $p(\Gamma_1, \tau, \mu) = p(\Gamma_1, \tau, \nu)$ and $p(\Gamma_2, \overline{\tau}, \mu) = p(\Gamma_2, \overline{\tau}, \nu)$, clusters μ and ν determine the same flip for $\Gamma_1(pert(\tau))$, a contradiction to the assumptions.

- Suppose next that Case 1 applies both to μ and ν to determine a different flip for $\Gamma_1(pert(\tau))$.

  By assumption, $\Gamma_1(pert(\tau))$ and $\Gamma_2(G(\overline{\tau}))$ are μ-sided. It follows that $\Gamma_1(G(\overline{\tau}))$ and $\Gamma_2(pert(\tau))$ are μ-sided, as well. In fact, they are not μ-traversable by Lemma 4.3. Also, if they are μ-kernelized, or μ-infeasible, or μ-bisided, then $H(\mu)$ would not be connected in Γ_1 or in Γ_2. This contradicts Lemma 4.1. Analogously, $\Gamma_1(pert(\tau))$, $\Gamma_2(G(\overline{\tau}))$, $\Gamma_1(G(\overline{\tau}))$, and $\Gamma_2(pert(\tau))$ are ν-sided.

  Assume, w.l.o.g. up to renaming $f'(\tau)$ with $f''(\tau)$ in Γ_i, that $p(\Gamma_i, \overline{\tau}, \nu)$ is incident to $f''(\tau)$, for $i = 1, 2$.

  Our strategy is to show that either $p(\Gamma_1, \tau, \mu) = p(\Gamma_1, \tau, \nu)$ and $p(\Gamma_2, \tau, \mu) = p(\Gamma_2, \tau, \nu)$ (Condition A) or that $p(\Gamma_1, \tau, \mu) \neq p(\Gamma_1, \tau, \nu)$ and $p(\Gamma_2, \tau, \mu) \neq p(\Gamma_2, \tau, \nu)$ (Condition B).

  We first show that, if Condition A or Condition B holds, then the assumption that ν and μ determine different flips for $\Gamma_1(pert(\tau))$ is contradicted. Suppose first that Condition A holds. Then, $p(\Gamma_1, \overline{\tau}, \mu) = p(\Gamma_1, \overline{\tau}, \nu)$ and $p(\Gamma_2, \overline{\tau}, \mu) = p(\Gamma_2, \overline{\tau}, \nu)$ and such paths are all incident to $f''(\tau)$, as otherwise $H(\mu)$ or $H(\nu)$ would not be connected in Γ_1 or in Γ_2, which is a contradiction by Lemma 4.1. Hence, the flip determined for $\Gamma_1(pert(\tau))$ by μ and ν is the same, a contradiction. Suppose next that Condition B holds. Then, $p(\Gamma_1, \overline{\tau}, \mu) \neq p(\Gamma_1, \overline{\tau}, \nu)$ and $p(\Gamma_2, \overline{\tau}, \mu) \neq p(\Gamma_2, \overline{\tau}, \nu)$, where $p(\Gamma_1, \overline{\tau}, \mu)$ and $p(\Gamma_2, \overline{\tau}, \mu)$ are incident to $f'(\tau)$, while $p(\Gamma_1, \overline{\tau}, \nu)$ and $p(\Gamma_2, \overline{\tau}, \nu)$ are incident to $f''(\tau)$, as otherwise $H(\mu)$ or $H(\nu)$ would not be connected in Γ_1 or in Γ_2, which is a contradiction by Lemma 4.1. Hence, the flip determined for $\Gamma_1(pert(\tau))$ by μ and ν is the same, a contradiction.

  We now prove that either Condition A or Condition B holds. Namely, we prove by contradiction that $p(\Gamma_1, \tau, \mu) = p(\Gamma_1, \tau, \nu)$ and $p(\Gamma_2, \tau, \mu) \neq p(\Gamma_2, \tau, \nu)$ do not hold simultaneously. The proof that $p(\Gamma_1, \tau, \mu) \neq p(\Gamma_1, \tau, \nu)$ and $p(\Gamma_2, \tau, \mu) = p(\Gamma_2, \tau, \nu)$ do not hold simultaneously is symmetrical. Since $p(\Gamma_i, \overline{\tau}, \nu)$ is incident to $f''(\tau)$, for $i = 1, 2$, we have that $p(\Gamma_1, \tau, \mu) = p(\Gamma_1, \tau, \nu)$ is incident to $f''(\tau)$ in Γ_1, as otherwise $H(\nu)$ would not be connected in Γ_1, that $p(\Gamma_1, \overline{\tau}, \mu)$ is incident to $f''(\tau)$ in Γ_1, as otherwise $H(\mu)$ would not be connected in Γ_1, that $p(\Gamma_2, \tau, \nu)$ is incident to $f''(\tau)$ in Γ_2, as otherwise $H(\nu)$ would not be connected in Γ_2, that $p(\Gamma_2, \tau, \mu)$ is incident to $f'(\tau)$ in Γ_1, since $p(\Gamma_2, \tau, \mu) \neq p(\Gamma_2, \tau, \nu)$, and that $p(\Gamma_2, \overline{\tau}, \mu)$ is incident to $f'(\tau)$ in Γ_2, as otherwise $H(\mu)$ would not be connected in Γ_2.

  We distinguish two cases.

Suppose that τ is an R-node. Denote by $V_\tau(\mu)$ and by $V_\tau(\nu)$ the set of vertices in $p(\Gamma_1, \tau, \mu) = p(\Gamma_1, \tau, \nu)$ belonging to μ and ν, respectively (see Fig. 4.12(a)). Since $p(\Gamma_2, \tau, \mu) \neq p(\Gamma_2, \tau, \nu)$ and since $\Gamma_2(pert(\tau))$ is μ-sided and ν-sided, it follows that none of the vertices in $V_\tau(\mu)$ is incident to $f''(\tau)$ in Γ_2 and none of the vertices in $V_\tau(\nu)$ is incident to $f'(\tau)$ in Γ_2. Since τ is an R-node, all the vertices in $V_\tau(\mu)$ are incident to internal faces of $\Gamma_2(pert(\tau))$ or all the vertices in $V_\tau(\nu)$ are incident to internal faces of $\Gamma_2(pert(\tau))$. Suppose that all the vertices in $V_\tau(\mu)$ are incident to internal faces of $\Gamma_2(pert(\tau))$, the other case being analogous (see Fig. 4.12(b)). Then, in order for $H(\mu)$ to be connected and for $\Gamma_2(pert(\tau))$ to be μ-sided, a child τ_i of τ in $\mathcal{T}$ that is incident to $f'(\tau)$ is μ-traversable in Γ_2. By Lemma 4.3, we have that τ_i is either μ-infeasible or μ-traversable in Γ_1. In the former case, a contradiction to the fact that Γ_1 is a $\langle 0, 0, \infty \rangle$-drawing of $C(G, T)$ is obtained. In the latter case, since τ_i is incident to $f'(\tau)$ in Γ_1, $\Gamma_1(pert(\tau))$ is not μ-sided, a contradiction.

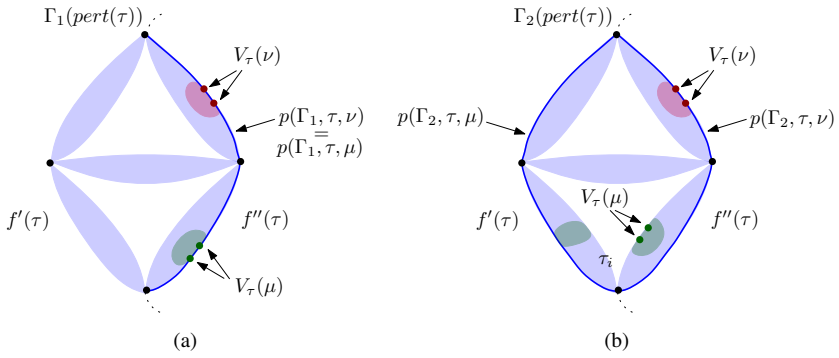


Figure 4.12: Illustration for the proof that $p(\Gamma_1, \tau, \mu) = p(\Gamma_1, \tau, \nu)$ and $p(\Gamma_2, \tau, \mu) \neq p(\Gamma_2, \tau, \nu)$ do not simultaneously hold if τ is an R-node. (a) Drawing Γ_1 with $p(\Gamma_1, \tau, \mu) = p(\Gamma_1, \tau, \nu)$. (b) Drawing Γ_2 with $p(\Gamma_2, \tau, \mu) \neq p(\Gamma_2, \tau, \nu)$.

Next, suppose that τ is a P-node. Consider the sequence $\tau_1, \tau_2, \ldots, \tau_p$ of children of τ that have vertices belonging to ν as they appear in $\Gamma_1(pert(\tau))$, where τ_1 is incident to $f''(\tau)$. Since $H(\nu)$ is connected in Γ_1, it follows that τ_i is ν-traversable in Γ_1, for $i = 1, \ldots, p-1$, and that τ_p is either ν-traversable or ν-sided. Analogous considerations hold for the sequence $\tau_1, \tau_2, \ldots, \tau_q$ of children of τ that have vertices belonging to μ. We distinguish two cases: Either $p \neq q$ (see Fig. 4.13(a)) or $p = q$ (see Fig. 4.13(b)).

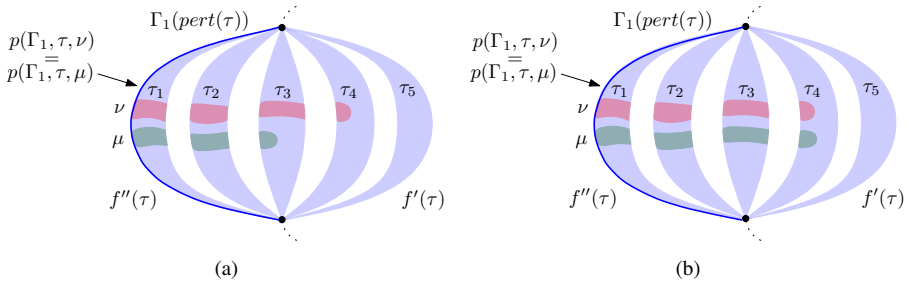


Figure 4.13: Illustration for the proof that $p(\Gamma_1,\tau,\mu) = p(\Gamma_1,\tau,\nu)$ and $p(\Gamma_2,\tau,\mu) \neq p(\Gamma_2,\tau,\nu)$ do not simultaneously hold if τ is a P-node. (a) The case $p > q$, with $p = 4$ and $q = 3$. (b) The case $p = q = 4$.

In the first case suppose, without loss of generality, that $p > q$. Then, for every $1 \leq i \leq q$, τ_i is ν-traversable in Γ_1. By Lemma 4.3, the embedding of $pert(\tau_i)$ in Γ_2 is either ν-infeasible or ν-traversable. In the former case, a contradiction to the fact that Γ_2 is a $\langle 0,0,\infty\rangle$-drawing of $C(G,T)$ is obtained. In the latter case, the child of τ that is incident to $f'(\tau)$ in Γ_2 is ν-traversable, thus contradicting the assumption that $\Gamma_2(pert(\tau))$ is ν-sided. Suppose next that $p = q$. If τ_p is not incident to $f'(\tau)$ in Γ_2, then the child of τ that is incident to $f'(\tau)$ in Γ_2 is one of $\tau_1,\tau_2,\ldots,\tau_{p-1}$, hence, by Lemma 4.3, it is either ν-infeasible in Γ_2, thus contradicting the fact that Γ_2 is a $\langle 0,0,\infty\rangle$-drawing of $C(G,T)$, or it is ν-traversable in Γ_2, thus contradicting the assumption that $\Gamma_2(pert(\tau))$ is ν-sided. Analogously, if τ_p is not incident to $f''(\tau)$ in Γ_2, then the child of τ that is incident to $f''(\tau)$ is one of $\tau_1,\tau_2,\ldots,\tau_{q-1}$, hence, by Lemma 4.3, it is either μ-infeasible in Γ_2, thus contradicting the fact that Γ_2 is a $\langle 0,0,\infty\rangle$-drawing of $C(G,T)$, or it is μ-traversable in Γ_2, thus contradicting the assumption that $\Gamma_2(pert(\tau))$ is μ-sided. We have a contradiction as τ_p can not be at the same time incident to both $f'(\tau)$ and $f''(\tau)$.

- Finally, suppose that Case 2 or Case 3 applies to μ and that Case 1 applies to ν to determine a different flip for $\Gamma_1(pert(\tau))$.

  We show how to restrict to the case in which μ is an ancestor of ν. First, if ν is an ancestor of μ, then $p(\Gamma_1,\tau,\mu)$ entirely belongs to ν, hence τ is ν-traversable, a contradiction to the fact that Case 1 applies to ν. Second, since Case 2 or Case 3 applies to μ, it follows that $p(\Gamma_1,\tau,\mu)$ and $p(\Gamma_2,\overline{\tau},\mu)$ are well-defined. Also, since Case 1 applies to ν, it follows that $p(\Gamma_1,\tau,\nu)$ and $p(\Gamma_2,\overline{\tau},\nu)$ are

also well-defined. Now suppose, for a contradiction, that μ is not an ancestor of ν. Since ν is not an ancestor of μ, it follows that μ and ν do not share vertices. Hence, $p(\Gamma_1, \tau, \mu) \neq p(\Gamma_1, \tau, \nu)$ and $p(\Gamma_2, \overline{\tau}, \mu) \neq p(\Gamma_2, \overline{\tau}, \nu)$. However, this implies that μ and ν determine the same flip for $\Gamma_1(pert(\tau))$, a contradiction to the assumptions. We can hence assume that μ is an ancestor of ν.

Since Case 1 applies to ν, we have that $\Gamma_1(pert(\tau))$ is ν-sided. It follows that $\Gamma_2(pert(\tau))$ is ν-sided, as well. Namely, it is not ν-traversable by Lemma 4.3. Also, if it is ν-kernelized, or ν-infeasible, or ν-bisided, then $H(\nu)$ would not be connected in Γ_2, given that $\Gamma_2(G(\overline{\tau}))$ is ν-sided and hence not ν-traversable. This contradicts Lemma 4.1. An analogous argument proves that $\Gamma_1(G(\overline{\tau}))$ is ν-sided.

– Suppose that Case 2 applies to μ. Then, $\Gamma_1(pert(\tau))$ and $\Gamma_2(G(\overline{\tau}))$ are both μ-side-spined and not μ-full. If $\Gamma_1(G(\overline{\tau}))$ is μ-central-spined, then Γ_1 contains a cycle whose vertices belong to μ containing in its interior a vertex in $G(\overline{\tau})$ not belonging to μ. This contradicts Lemma 4.1. It follows that $\Gamma_1(G(\overline{\tau}))$ is μ-side-spined and not μ-full. An analogous argument proves that $\Gamma_2(pert(\tau))$ is μ-side-spined and not μ-full.

  Assume, without loss of generality up to renaming $f'(\tau)$ with $f''(\tau)$ that $p(\Gamma_i, \overline{\tau}, \mu)$ is incident to $f''(\tau)$ in Γ_i, for $i = 1, 2$. Then, for $i = 1, 2$, path $p(\Gamma_i, \tau, \mu)$ is incident to $f''(\tau)$ in Γ_i, as otherwise $p(\Gamma_i, \overline{\tau}, \mu)$ together with $p(\Gamma_i, \tau, \mu)$ forms a cycle that entirely belongs to μ and that contains in its interior a vertex not belonging to μ, which by Lemma 4.1 contradicts the assumption that Γ_i is a $\langle 0, 0, \infty \rangle$-drawing of $C(G, T)$.

  Our strategy is to show that either $p(\Gamma_1, \tau, \mu) = p(\Gamma_1, \tau, \nu)$ and $p(\Gamma_2, \tau, \mu) = p(\Gamma_2, \tau, \nu)$ (Condition A) or that $p(\Gamma_1, \tau, \mu) \neq p(\Gamma_1, \tau, \nu)$ and $p(\Gamma_2, \tau, \mu) \neq p(\Gamma_2, \tau, \nu)$ (Condition B).

  We first show that, if Condition A or Condition B holds, then the assumption that ν and μ determine different flips for $\Gamma_1(pert(\tau))$ is contradicted. Suppose first that Condition A holds. Then, for $i = 1, 2$, we have that $p(\Gamma_i, \overline{\tau}, \mu) = p(\Gamma_i, \overline{\tau}, \nu)$, as otherwise $H(\nu)$ would not be connected in Γ_i, which is a contradiction by Lemma 4.1. Hence, the flip determined for $\Gamma_1(pert(\tau))$ by μ and ν is the same, a contradiction. Suppose next that Condition B holds. Then, for $i = 1, 2$, we have that $p(\Gamma_i, \overline{\tau}, \mu) \neq p(\Gamma_i, \overline{\tau}, \nu)$, as otherwise $H(\nu)$ would not be connected in Γ_i, which is a contradiction by Lemma 4.1. Hence, the flip determined for $\Gamma_1(pert(\tau))$ by μ and ν is the same, a contradiction.

We now prove that either Condition A or Condition B holds. Namely, we prove by contradiction that $p(\Gamma_1, \tau, \mu) = p(\Gamma_1, \tau, \nu)$ and $p(\Gamma_2, \tau, \mu) \neq p(\Gamma_2, \tau, \nu)$ do not simultaneously hold. The proof that $p(\Gamma_1, \tau, \mu) \neq p(\Gamma_1, \tau, \nu)$ and $p(\Gamma_2, \tau, \mu) = p(\Gamma_2, \tau, \nu)$ do not simultaneously hold is symmetrical. Hence, $p(\Gamma_1, \tau, \mu) = p(\Gamma_1, \tau, \nu)$ is incident to $f''(\tau)$ in Γ_1, $p(\Gamma_1, \overline{\tau}, \nu)$ is incident to $f''(\tau)$ in Γ_1, as otherwise $H(\nu)$ would not be connected in Γ_1, $p(\Gamma_2, \tau, \nu) \neq p(\Gamma_2, \tau, \mu)$ is incident to $f'(\tau)$ in Γ_2, and $p(\Gamma_2, \overline{\tau}, \nu)$ is incident to $f'(\tau)$ in Γ_2, as otherwise $H(\nu)$ would not be connected in Γ_2.

We distinguish two cases.

Suppose that τ is an R-node. Denote by $V_\tau(\nu)$ the set of vertices in $p(\Gamma_1, \tau, \mu) = p(\Gamma_1, \tau, \nu)$ belonging to ν. Since $p(\Gamma_2, \tau, \mu) \neq p(\Gamma_2, \tau, \nu)$ and since $\Gamma_2(pert(\tau))$ is ν-sided, it follows that none of the vertices in $V_\tau(\nu)$ is incident to $f''(\tau)$ in Γ_2. Hence, all the vertices in $V_\tau(\nu)$ are incident to internal faces of $\Gamma_2(pert(\tau))$. Then, in order for $H(\nu)$ to be connected and for $\Gamma_2(pert(\tau))$ to be ν-sided, a child τ_i of τ in $\mathcal{T}$ that is incident to $f'(\tau)$ in Γ_2 is ν-traversable. By Lemma 4.3, we have that τ_i is either μ-infeasible or μ-traversable in Γ_1. In the former case, a contradiction to the fact that Γ_1 is a $\langle 0, 0, \infty \rangle$-drawing of $C(G, T)$ is obtained. In the latter case, since τ is an R-node, τ_i is incident to $f'(\tau)$ in Γ_1. This contradicts the fact that $\Gamma_1(pert(\tau))$ is ν-sided.

Next, suppose that τ is a P-node. Let k be the number of children of τ in $\mathcal{T}$. Consider the sequence $\tau_1, \tau_2, \ldots, \tau_p$ of children of τ that have vertices belonging to ν as they appear in $\Gamma_1(pert(\tau))$, where τ_1 is incident to $f''(\tau)$. Since $H(\nu)$ is connected in Γ_1, it follows that τ_i is ν-traversable in Γ_1, for $i = 1, \ldots, p-1$, and that τ_p is either ν-traversable or ν-sided. Also, consider the sequence $\tau_1, \tau_2, \ldots, \tau_q$ of children of τ that are μ-spined, as they appear in $\Gamma_1(pert(\tau))$, where τ_1 is incident to $f''(\tau)$. Since $\Gamma_1(pert(\tau))$ is μ-side-spined, it follows that τ_i is μ-full, for $i = 1, \ldots, q-1$, and that τ_q is either μ-full or μ-side-spined. Observe that $p, q \geq 1$, since $\Gamma_1(pert(\tau))$ is ν-sided and μ-side-spined. Also observe that $q < k$ or τ_q is not μ-full, as otherwise τ would be μ-full, which contradicts the assumptions.

We distinguish some cases.

* Suppose first that $p \leq q < k$. Since $\Gamma_2(pert(\tau))$ is μ-side-spined and $p(\Gamma_2, \tau, \mu)$ is incident to $f''(\tau)$ in Γ_2, it follows that $\tau_1, \tau_2, \ldots, \tau_q$ are the first q children of τ as they appear in $\Gamma_2(pert(\tau))$, possibly in a different relative order with respect to their order in $\Gamma_1(pert(\tau))$,

where one of $\tau_1, \tau_2, \ldots, \tau_q$ is incident to $f''(\tau)$. Hence, the child of τ incident to $f'(\tau)$ does not contain any vertex belonging to ν, which contradicts either $p(\Gamma_2, \tau, \mu) \neq p(\Gamma_2, \tau, \nu)$ or the fact that $\Gamma_2(pert(\tau))$ is ν-sided.

- Suppose next that $p = q = k$. Then, τ_q is not μ-full. Since $\Gamma_2(pert(\tau))$ is μ-side-spined and $p(\Gamma_2, \tau, \mu)$ is incident to $f''(\tau)$ in Γ_2, it follows that τ_q is the child of τ incident to $f'(\tau)$ in Γ_2. Thus, by Lemma 4.3, the child of τ incident to $f''(\tau)$ in Γ_2 is either ν-infeasible, thus contradicting the fact that Γ_2 is a $\langle 0, 0, \infty \rangle$-drawing of $C(G, T)$, or it is ν-traversable, which contradicts either $p(\Gamma_2, \tau, \mu) \neq p(\Gamma_2, \tau, \nu)$ or the fact that $\Gamma_2(pert(\tau))$ is ν-sided.
- Suppose next that $p < q = k$. Then, τ_q is not μ-full. Since $\Gamma_2(pert(\tau))$ is μ-side-spined and $p(\Gamma_2, \tau, \mu)$ is incident to $f''(\tau)$ in Γ_2, it follows that τ_q is the child of τ incident to $f'(\tau)$ in Γ_2. Hence, no vertex in ν is incident to $f'(\tau)$ in Γ_2, which contradicts either $p(\Gamma_2, \tau, \mu) \neq p(\Gamma_2, \tau, \nu)$ or the fact that $\Gamma_2(pert(\tau))$ is ν-sided.
- Suppose finally that $p > q$. Then, all of $\tau_1, \tau_2, \ldots, \tau_q$ are ν-traversable. Since $\tau_1, \tau_2, \ldots, \tau_q$ are the first q children of τ as they appear in $\Gamma_2(pert(\tau))$, possibly in a different relative order, where one of $\tau_1, \tau_2, \ldots, \tau_q$ is incident to $f''(\tau)$, by Lemma 4.3 and since no child of τ is ν-infeasible in Γ_2, as otherwise Γ_2 would not be a $\langle 0, 0, \infty \rangle$-drawing of $C(G, T)$, it follows that $\Gamma_2(pert(\tau))$ has a vertex belonging to ν and incident to $f''(\tau)$, thus contradicting the assumption that $p(\Gamma_1, \tau, \nu)$ is incident to $f''(\tau)$ or the assumption that $\Gamma_1(pert(\tau))$ is ν-sided.

– Suppose that Case 3 applies to μ. Then, $G(\overline{\tau})$ is μ-full. By assumption, $\Gamma_1(pert(\tau))$ is μ-side-spined and not μ-full. If at least one of the poles of τ is not incident to the outer face of Γ_1, then Γ_1 contains a cycle in $G(\overline{\tau})$ whose vertices belong to μ containing in its interior a vertex in $pert(\tau)$ not belonging to μ. This contradicts Lemma 4.1. Assume then that both poles of τ are incident to the outer face of Γ_1. If $p(\Gamma_1, \tau, \mu)$ delimits the outer face of Γ_1, then $p(\Gamma_1, \tau, \mu)$ together with $p(\Gamma_1, \overline{\tau}, \mu)$ forms a cycle whose vertices belong to μ containing in its interior a vertex in $pert(\tau)$ not belonging to μ, again contradicting the assumption that Γ_1 is a $\langle 0, 0, \infty \rangle$-drawing of $C(G, T)$, by Lemma 4.1. Also, recall that, by assumption, both the poles of τ are incident to the outer face of Γ_2.

Assume, without loss of generality up to renaming $f'(\tau)$ with $f''(\tau)$ with that $p(\Gamma_i, \overline{\tau}, \mu)$ is incident to $f''(\tau)$ in Γ_i, for $i = 1, 2$. Hence, $f'(\tau)$ is

the outer face of Γ_i, for $i = 1, 2$, and $p(\Gamma_i, \overline{\tau}, \mu)$ delimits $f''(\tau)$ in Γ_i, for $i = 1, 2$, by definition.

The reminder of the proof is exactly the same as when Case 2 applies to μ.

This concludes the proof of the claim. □

By Claim 4.3, no two clusters $\mu \neq \nu \in T$ exist that determine different flips for $\Gamma_1(pert(\tau))$ when constructing Γ_3. Then, by Claims 4.1 and 4.2, the constructed embedding Γ_3 of $C(G,T)$ satisfies Conditions (i) and (ii) of Lemma 4.1, for each cluster μ, hence Γ_3 is a $\langle 0, 0, \infty \rangle$-drawing. This concludes the proof of Lemma 4.4.

We now determine conditions for the nodes of the SPQR-tree $\mathcal{T}$ of G in order for $C(G,T)$ to admit a $\langle 0, 0, \infty \rangle$-drawing.

We say that an embedding $\Gamma(sk(\tau))$ of $sk(\tau)$ in which the edge e representing the parent of τ is incident to the outer face is *extensible* if the following condition holds: If $C(G,T)$ admits a $\langle 0, 0, \infty \rangle$-drawing in which e_ρ is incident to the outer face, then it admits a $\langle 0, 0, \infty \rangle$-drawing in which e_ρ is incident to the outer face and in which the embedding of $sk(\tau)$ is $\Gamma(sk(\tau))$.

For an embedding $\Gamma(sk(\tau))$ of $sk(\tau)$ and a cluster μ in T, we define an auxiliary graph $G'(\tau, \mu)$ as follows. Graph $G'(\tau, \mu)$ has one vertex v_f for each face f of $\Gamma(sk(\tau))$ containing a μ-traversable virtual edge on its boundary; two vertices of $G'(\tau, \mu)$ are connected by an edge if they share a μ-traversable virtual edge. We are now ready to prove the following main lemma.

Lemma 4.5 *Let $C(G,T)$ be a clustered graph, with G biconnected, that admits a $\langle 0, 0, \infty \rangle$-drawing in which the edge e_ρ representing the root ρ of the SPQR-tree $\mathcal{T}$ of G is incident to the outer face. Then, an embedding $\Gamma(sk(\tau))$ in which the edge e representing the parent of τ is incident to the outer face is extensible if and only if the following properties hold. For each cluster $\mu \in T$:*

- *(A) There exists no cycle in $\Gamma(sk(\tau))$ that is composed of μ-spined virtual edges $e_1, \ldots, e_h$ containing in its interior a virtual edge that is not μ-full;*
- *(B) $G'(\tau, \mu)$ is connected; and*
- *(C) if $G'(\tau, \mu)$ contains at least one vertex, then each virtual edge of $sk(\tau)$ which is μ-touched and not μ-traversable shares a face with a μ-traversable virtual edge in $\Gamma(sk(\tau))$. Otherwise (that is, if $G'(\tau, \mu)$ contains no vertex), all the μ-touched virtual edges are incident to the same face of $\Gamma(sk(\tau))$.*

In the following we prove Lemma 4.5. We first prove the necessity. Suppose that an embedding $\Gamma(sk(\tau))$ of $sk(\tau)$ is extensible. That is, $C(G,T)$ admits a $\langle 0,0,\infty\rangle$-drawing in which e_ρ is incident to the outer face and in which the embedding of $sk(\tau)$ is $\Gamma(sk(\tau))$. We prove that $\Gamma(sk(\tau))$ satisfies Properties (A), (B), and (C) by means of suitable claims.

Claim 4.4 $\Gamma(sk(\tau))$ *satisfies Property (A).*

Proof: No cycle $(e_1,\ldots,e_h)$ in $\Gamma(sk(\tau))$ such that virtual edges $e_1,\ldots,e_h$ are μ-spined contains in its interior a virtual edge that is not μ-full, as otherwise, in any drawing Γ of $C(G,T)$ in which e_ρ is incident to the outer face and in which the embedding of $sk(\tau)$ is $\Gamma(sk(\tau))$, there would exist a cycle whose vertices all belong to μ enclosing a vertex not belonging to μ, thus implying that $R(\mu)$ is not simple or that Γ contains an edge-region crossing. □

Claim 4.5 $\Gamma(sk(\tau))$ *satisfies Property (B).*

Proof: We have that $G'(\tau,\mu)$ is connected, as otherwise, in any drawing Γ of $C(G,T)$ in which e_ρ is incident to the outer face and in which the embedding of $sk(\tau)$ is $\Gamma(sk(\tau))$, there would exist a cycle $\mathcal{C}$ such that none of the vertices of $\mathcal{C}$ belongs to μ and such that $\mathcal{C}$ separates vertices belonging to μ, thus implying that $R(\mu)$ is not simple or that Γ contains an edge-region crossing. □

Claim 4.6 $\Gamma(sk(\tau))$ *satisfies Property (C).*

Proof: We distinguish the case in which $G'(\tau,\mu)$ contains at least one vertex and the case in which $G'(\tau,\mu)$ contains no vertex.

- Suppose that $G'(\tau,\mu)$ contains at least one vertex. Refer to Fig. 4.14(a). Then, we prove that each virtual edge of $sk(\tau)$ which is μ-touched and not μ-traversable shares a face with a μ-traversable virtual edge in $\Gamma(sk(\tau))$.

  Suppose, for a contradiction, that there exists a virtual edge e of $sk(\tau)$ that is μ-touched, that is not μ-traversable, and that does not share any face with a μ-traversable virtual edge in $\Gamma(sk(\tau))$. Consider the two faces f_e^1 and f_e^2 of $\Gamma(sk(\tau))$ incident to e. Denote by $\mathcal{C}_s$ the cycle of virtual edges composed of the edges delimiting f_e^1 and f_e^2, except for e.

  By assumption, for each edge e_i of $\mathcal{C}_s$, the node τ_i of $\mathcal{T}$ corresponding to e_i is such that any embedding of $pert(\tau_i)$ is not μ-traversable. Also, in any $\langle 0,0,\infty\rangle$-drawing of $C(G,T)$, the embedding of $pert(\tau_i)$ is not μ-infeasible.

Hence, by Lemma 4.2, there exists a path in $pert(\tau_i)$ that connects the poles of τ_i, that is different from the edge connecting the poles of τ_i, and none of whose vertices belongs to μ. Concatenating all such paths for all the nodes of $\mathcal{T}$ corresponding to the edges of $\mathcal{C}_s$ results in a cycle $\mathcal{C}$ such that none of the vertices of $\mathcal{C}$ belongs to μ and such that $\mathcal{C}$ passes through all the vertices of $\mathcal{C}_s$. Observe that $\mathcal{C}$ contains vertices of μ on both sides, namely vertices of μ in $pert(e)$ and vertices of μ in the pertinent graph of a virtual edge e' of $sk(\tau)$ which is μ-traversable; such an edge e' exists since $G'(\tau, \mu)$ contains at least one vertex. Hence, in any drawing Γ of $C(G, T)$ in which the embedding of $sk(\tau)$ is $\Gamma(sk(\tau))$, there exists a cycle $\mathcal{C}$ such that none of the vertices of $\mathcal{C}$ belongs to μ and such that $\mathcal{C}$ separates vertices belonging to μ, thus implying that $R(\mu)$ is not simple or that Γ contains an edge-region crossing, a contradiction.

- Suppose that $G'(\tau, \mu)$ contains no vertex. Refer to Fig. 4.14(b). We prove that all the μ-touched virtual edges are incident to the same face of $\Gamma(sk(\tau))$.

  Suppose, for a contradiction, that there exists no face of $\Gamma(sk(\tau))$ such that all the μ-touched virtual edges of $\Gamma(sk(\tau))$ are incident to such a face. Consider the two faces f_e^1 and f_e^2 of $\Gamma(sk(\tau))$ incident to any μ-touched virtual edge e. Denote by $\mathcal{C}_s^1$ and $\mathcal{C}_s^2$ the cycles delimiting f_e^1 and f_e^2, respectively.

  By assumption, for each edge e_i of $\mathcal{C}_s^1$ (of $\mathcal{C}_s^2$), the node τ_i of $\mathcal{T}$ corresponding to e_i is such that any embedding of $pert(\tau_i)$ is not μ-traversable. Also, in any $\langle 0, 0, \infty\rangle$-drawing of $C(G, T)$, the embedding of $pert(\tau_i)$ is not μ-infeasible. Hence, by Lemma 4.2, there exists a path in $pert(\tau_i)$ that connects the poles of τ_i, that is different from the edge connecting the poles of τ_i, and none of whose vertices belongs to μ. Concatenating all such paths for all the nodes of $\mathcal{T}$ corresponding to the edges of $\mathcal{C}_s^1$ (resp. of $\mathcal{C}_s^2$) results in a cycle $\mathcal{C}^1$ (resp. $\mathcal{C}^2$) such that none of the vertices of $\mathcal{C}^1$ (resp. of $\mathcal{C}^2$) belongs to μ and such that $\mathcal{C}^1$ (resp. $\mathcal{C}^2$) passes through all the vertices of $\mathcal{C}_s^1$ (resp. $\mathcal{C}_s^2$). Then, $\mathcal{C}^1$ or $\mathcal{C}^2$ contains vertices of μ on both sides, namely vertices of μ in $pert(e)$ and vertices of μ in the pertinent graph of a virtual edge e' of $sk(\tau)$ which is not incident to f_e^1 or to f_e^2, respectively; such an edge e' exists since not all the μ-touched virtual edges are incident to the same face of $\Gamma(sk(\tau))$. This implies that $R(\mu)$ is not simple or that Γ contains an edge-region crossing.

This concludes the proof of the claim. □

We now prove the sufficiency. Namely, suppose that Properties (A), (B), and (C) hold for an embedding $\Gamma(sk(\tau))$ of $sk(\tau)$. We prove that $\Gamma(sk(\tau))$ is extensible, that is, we show that a $\langle 0, 0, \infty\rangle$-drawing of $C(G, T)$ exists in which e_ρ is incident to the outer face and in which the embedding of $sk(\tau)$ is $\Gamma(sk(\tau))$.

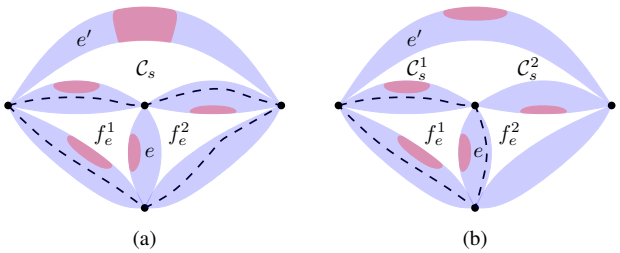


Figure 4.14: Proof that Property (C) is satisfied when $C(G,T)$ admits a $\langle 0,0,\infty\rangle$-drawing in which e_ρ is incident to the outer face and in which the embedding of $sk(\tau)$ is $\Gamma(sk(\tau))$. (a) $G'(\tau,\mu)$ contains at least one vertex and (b) $G'(\tau,\mu)$ contains no vertex. In both figures, cycle $\mathcal{C}$ is represented by a dashed curve.

Let Γ' be any $\langle 0,0,\infty\rangle$-drawing of $C(G,T)$ in which e_ρ is incident to the outer face (clustered graph $C(G,T)$ admits such a drawing by hypothesis). Let $\Gamma'(sk(\tau))$ be the embedding of $sk(\tau)$ in Γ'. If $\Gamma'(sk(\tau))$ coincides with $\Gamma(sk(\tau))$, then there is nothing to prove. Otherwise, assume that the two embeddings of $sk(\tau)$ do not coincide. Observe that this implies that τ is not an S-node, as the skeleton of an S-node is a cycle, which has a unique embedding.

Next, suppose that τ is an R-node. Then, since $sk(\tau)$ has exactly two embeddings, which are one the flip of the other, $\Gamma(sk(\tau))$ is the flip of $\Gamma'(sk(\tau))$. Consider the drawing Γ of $C(G,T)$ obtained by flipping Γ' around the poles of the root ρ of $\mathcal{T}$, that is, by reverting the adjacency list of every vertex of $C(G,T)$. Observe that Γ is a $\langle 0,0,\infty\rangle$-drawing since Γ' is. Also, e_ρ is incident to the outer face of Γ since it is incident to the outer face of Γ'. Moreover, the embedding of $sk(\tau)$ in Γ is the flip of the embedding $\Gamma'(sk(\tau))$ of $sk(\tau)$ in Γ', hence it coincides with $\Gamma(sk(\tau))$. Thus a $\langle 0,0,\infty\rangle$-drawing Γ of $C(G,T)$ in which e_ρ is incident to the outer face and in which the embedding of $sk(\tau)$ is $\Gamma(sk(\tau))$ exists.

It remains to consider the case in which τ is a P-node. We show how to construct a $\langle 0,0,\infty\rangle$-drawing Γ^C in which e_ρ is incident to the outer face and in which the embedding of $sk(\tau)$ is $\Gamma(sk(\tau))$. For every neighbor τ' of τ in $\mathcal{T}$ (including its parent), denote by $\Gamma^1(pert(\tau'))$ the embedding of $pert(\tau')$ in Γ'. Moreover, denote by $\Gamma^2(pert(\tau'))$ the embedding of $pert(\tau')$ obtained by flipping $\Gamma^1(pert(\tau'))$ around the poles of τ'. Drawing Γ^C is such that, for every neighbor τ' of τ in $\mathcal{T}$, the embedding of $pert(\tau')$ is either $\Gamma^1(pert(\tau'))$ or $\Gamma^2(pert(\tau'))$.

The remainder of the proof is devoted to the P-node case and is structured into three parts as follows.

In the first part, we describe how to obtain Γ^C. In particular, we show how to choose the embedding of $pert(\tau')$ to be either $\Gamma^1(pert(\tau'))$ or $\Gamma^2(pert(\tau'))$ based on the constraints imposed by the clusters containing vertices of $pert(\tau')$.

In the second part, we show that the algorithm we describe in the first part univocally determines the embedding of $pert(\tau')$ to be either $\Gamma^1(pert(\tau'))$ or $\Gamma^2(pert(\tau'))$.

In the third part, we show that the performed choices actually lead to a $\langle 0, 0, \infty \rangle$-drawing of $C(G, T)$ in which e_ρ is incident to the outer face and in which the embedding of $sk(\tau)$ is $\Gamma(sk(\tau))$.

First part. We show an algorithm to determine Γ^C. We fix the embedding of $sk(\tau)$ to be $\Gamma(sk(\tau))$ with an outer face that is any of the two faces incident to the virtual edge of $sk(\tau)$ representing the parent of τ in $\mathcal{T}$. Later, we will possibly modify the choice of the outer face. We now show how to choose the embedding of $pert(\tau')$. This is done according to rules that aim at satisfying Conditions (i) and (ii) of Lemma 4.1. Namely, for each cluster $\mu \in T$ apply one of the following rules.

- *Rules to satisfy Condition (i) of Lemma 4.1.*

  Consider any neighbor τ' of τ in $\mathcal{T}$ such that $\Gamma^1(pert(\tau'))$ is μ-sided. Also, consider the neighbors τ'' and τ''' of τ in $\mathcal{T}$ following and preceding τ' in the circular order of the neighbors of τ determined by $\Gamma(sk(\tau))$, respectively. Without loss of generality assume that if the embedding of $pert(\tau')$ is $\Gamma^1(pert(\tau'))$, then a vertex in $pert(\tau')$ and in μ is incident to the face of $\Gamma(sk(\tau))$ to which τ'' is incident; and if the embedding of $pert(\tau')$ is $\Gamma^2(pert(\tau'))$, then a vertex in $pert(\tau')$ and in μ is incident to the face of $\Gamma(sk(\tau))$ to which τ''' is incident.

  - **Rule RI-1** If τ'' is μ-traversable and τ''' is not (if τ''' is μ-traversable and τ'' is not), then choose the embedding of $pert(\tau')$ to be $\Gamma^1(pert(\tau'))$ (resp. $\Gamma^2(pert(\tau'))$), see Fig. 4.15(a).
  - **Rule RI-2** If none of τ'' and τ''' is μ-traversable, if τ'' is μ-sided and τ''' is not (if τ''' is μ-sided and τ'' is not), then choose the embedding of $pert(\tau')$ to be $\Gamma^1(pert(\tau'))$ (resp. $\Gamma^2(pert(\tau'))$), see Fig. 4.15(b).

- *Rules to satisfy Condition (ii) of Lemma 4.1.*

  Consider any neighbor τ' of τ in $\mathcal{T}$ such that $\Gamma^1(pert(\tau'))$ is μ-side-spined and not μ-full. Also, consider the neighbors τ'' and τ''' of τ in $\mathcal{T}$ following and preceding τ' in the circular order of the neighbors of τ determined by $\Gamma(sk(\tau))$, respectively. Without loss of generality assume that if the embedding of $pert(\tau')$ is $\Gamma^1(pert(\tau'))$, then the path $P(\mu)$ delimiting the outer face of $\Gamma^1(pert(\tau'))$ and composed only of vertices in μ is incident to the face of $\Gamma(sk(\tau))$ to which

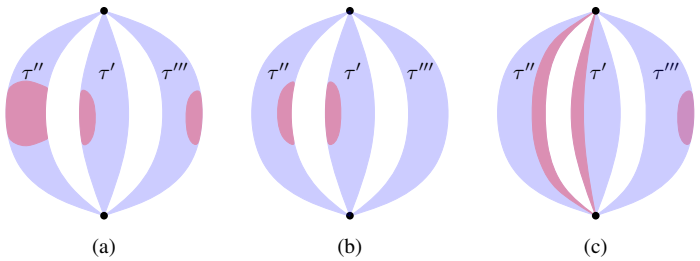


Figure 4.15: Choosing the embedding of $pert(\tau')$ in order to satisfy Conditions (i) and (ii) of Lemma 4.1 when τ is a P-node. (a) If τ'' is μ-traversable and τ''' is not, then the embedding of $pert(\tau')$ is $\Gamma^1(pert(\tau'))$; (b) if none of τ'' and τ''' is μ-traversable and if τ'' is μ-sided, then the embedding of $pert(\tau')$ is $\Gamma^1(pert(\tau'))$; (c) if τ'' is μ-spined, then the embedding of $pert(\tau')$ is $\Gamma^1(pert(\tau'))$.

τ'' is incident; and if the embedding of $pert(\tau')$ is $\Gamma^2(pert(\tau'))$, then $P(\mu)$ is incident to the face of $\Gamma(sk(\tau))$ to which τ''' is incident.

- **Rule RII-1** If τ'' is μ-spined and τ''' is not (resp. if τ''' is μ-spined and τ'' is not), and the face shared by τ' and τ'' in $\Gamma(sk(\tau))$ (τ' and τ''' in $\Gamma(sk(\tau))$) is different from the outer face of $\Gamma(sk(\tau))$, then choose the embedding of $pert(\tau')$ to be $\Gamma^1(pert(\tau'))$ (resp. $\Gamma^2(pert(\tau'))$), see Fig. 4.15(c).

- Suppose that the embedding of $pert(\tau')$, for some neighbor τ' of τ, has not been determined by rules RI-1, RI-2, and RII-1 over all clusters $\mu \in T$.

  - **Rule R0** If τ' is the parent of τ in $\mathcal{T}$, then set the embedding of $pert(\tau')$ to be $\Gamma^1(pert(\tau'))$ or $\Gamma^2(pert(\tau'))$ so that e_ρ is incident to the outer face. Otherwise, arbitrarily set the embedding of $pert(\tau')$ to be $\Gamma^1(pert(\tau'))$ or $\Gamma^2(pert(\tau'))$.

Denote by Γ the drawing constructed by the described algorithm. In order to complete the construction of Γ^C, we (possibly) modify the outer face of Γ. Namely, if e_ρ is incident to the outer face of Γ, then we let $\Gamma^C = \Gamma$. Otherwise, we choose as outer face of Γ^C the face of Γ that is incident to e_ρ and that is delimited by paths belonging to the pertinent graphs of different neighbors of τ in $\mathcal{T}$. Observe that one of such neighbors is the parent of τ in $\mathcal{T}$. Also, the choice of the outer face of Γ^C does

not alter the embedding $\Gamma(sk(\tau))$ of $sk(\tau)$. Namely, a different outer face is chosen as outer face in $\Gamma(sk(\tau))$, however the circular ordering of the virtual edges around the poles stays the same; moreover, the virtual edge representing the parent of τ in $\mathcal{T}$ is still incident to the outer face of $\Gamma(sk(\tau))$ in Γ^C.

Second part. We now prove that the embedding choices performed when considering two distinct clusters do not conflict.

Claim 4.7 *For any two clusters μ and ν ($\mu \neq \nu$), the application of the above rules when μ is considered does not produce an embedding choice for $pert(\tau')$ that is conflicting with the one that is produced when ν is considered.*

Proof: The proof is independent of the modification of the outer face of Γ to obtain Γ^C, hence we will refer to drawing Γ rather than to Γ^C. The proof distinguishes several cases.

Case 1: Suppose that the embedding of $pert(\tau')$ has been determined to be $\Gamma^1(pert(\tau'))$ by Rule RI-1 because τ' is μ-sided, because τ'' is μ-traversable, and because τ''' is not μ-traversable, for some cluster $\mu \in T$, or by Rule RI-2 because τ' is μ-sided, because τ'' is μ-sided, and because τ''' is not μ-traversable and not μ-sided.

Case 1A: Suppose, for a contradiction, that the embedding of $pert(\tau')$ has been determined to be $\Gamma^2(pert(\tau'))$ by Rule RI-1 because τ' is ν-sided, because τ''' is ν-traversable, and because τ'' is not ν-traversable, or by Rule RI-2 because τ' is ν-sided, because τ''' is ν-sided, and because τ'' is not ν-traversable and not ν-sided, for some cluster $\nu \in T$ with $\nu \neq \mu$.

We have that Condition (i) of Lemma 4.1 is not satisfied by Γ', a contradiction. Namely, one of the two paths between the poles of τ' delimiting the outer face of $\Gamma^1(pert(\tau'))$, say $P(\mu, \nu)$, contains vertices in μ and in ν, while the other path does not. Hence, if Γ' is such that τ'' is found before τ''' when traversing the neighbors of τ in $\mathcal{T}$ starting from τ' in the direction "defined by $P(\mu, \nu)$", then $H(\nu)$ is not connected, given that τ' and τ'' are not ν-traversable. Otherwise $H(\mu)$ is not connected, given that τ' and τ''' are not μ-traversable.

Case 1B: Suppose, for a contradiction, that the embedding of $pert(\tau')$ has been determined by Rule RII-1 to be $\Gamma^2(pert(\tau'))$ because τ' is ν-side-spined and not ν-full, and because τ''' is ν-spined and τ'' is not, for some cluster $\nu \in T$ with $\nu \neq \mu$. Then one of the two paths between the poles of τ' delimiting the outer face of $\Gamma^1(pert(\tau'))$, say $P(\nu)$, entirely belongs to ν and the same path also contains a vertex in μ.

This gives rise to a contradiction if μ is not a descendant of ν in T. Hence, assume that μ is a descendant of ν in T. If Γ' is such that τ''' is found before τ'' when

traversing the neighbors of τ in $\mathcal{T}$ starting from τ' in the direction "defined by $P(\nu)$", then $H(\mu)$ is not connected, given that τ' and τ''' are not μ-traversable, thus Condition (i) of Lemma 4.1 is not satisfied by Γ', a contradiction. Otherwise, there exists a cycle in Γ' whose vertices belong to ν and whose interior contains in Γ' a vertex not belonging to ν, thus implying that Condition (ii) of Lemma 4.1 is not satisfied by Γ', a contradiction. Namely, such a cycle is composed of $P(\nu)$ and of any path belonging to ν and connecting the poles of τ''' in $pert(\tau''')$; the vertex not in ν in the interior of this cycle is either a vertex in $pert(\tau')$ not in ν (which exists since τ' is not ν-full), or a vertex in $pert(\tau'')$ not in ν (which exists since τ'' is not ν-spined and hence not ν-full), depending on the "position" of the outer face in Γ'.

Case 2: Suppose that the embedding of $pert(\tau')$ has been determined to be $\Gamma^1(pert(\tau'))$ by Rule RII-1 because τ' is μ-side-spined and not μ-full, and because τ'' is μ-spined and τ''' is not, for some cluster $\mu \in T$.

Case 2A: Suppose that the embedding of $pert(\tau')$ has been determined to be $\Gamma^2(pert(\tau'))$ by Rule RI-1 (by Rule RI-2) because τ' is ν-sided, because τ''' is ν-traversable (ν-sided, respectively), and because τ'' is not ν-traversable (not ν-traversable and not ν-sided), for some cluster $\nu \in T$ with $\nu \neq \mu$. Such a case can be discussed analogously to Case 1B.

Case 2B: Suppose that the embedding of $pert(\tau')$ has been determined to be $\Gamma^2(pert(\tau'))$ by Rule RII-1 because τ' is ν-side-spined and not ν-full, and because τ''' is ν-spined and τ'' is not, for some cluster $\nu \in T$ with $\nu \neq \mu$. Observe that, since μ and ν share vertices, one of them is the ancestor of the other one. Assume without loss of generality that μ is an ancestor of ν. Since τ''' is ν-spined, it is also μ-spined, a contradiction.

This concludes the proof of the claim. □

Third part. We now prove that the drawing Γ^C resulting from the above described algorithm is a $\langle 0, 0, \infty \rangle$-drawing of $C(G, T)$ by exploiting Lemma 4.1. Observe that edge e_ρ is incident to the outer face of Γ^C by construction.

Claim 4.8 *Drawing Γ^C satisfies Condition (i) of Lemma 4.1.*

Proof: Consider any cluster μ. Observe that, by Lemma 4.1, $H(\mu)$ is connected in Γ' since Γ' is a $\langle 0, 0, \infty \rangle$-drawing. We prove that $H(\mu)$ is connected in Γ^C. Observe that $H(\mu)$ is connected in Γ^C if and only if it is connected in Γ. Hence, we will refer to drawing Γ rather than to Γ^C. Suppose, for a contradiction, that $H(\mu)$ is not connected in Γ.

For any neighbor τ' of τ in $\mathcal{T}$, we can assume that $\Gamma^1(pert(\tau'))$ is neither μ-infeasible, nor μ-kernelized, nor μ-bisided. Namely:

- If $\Gamma^1(pert(\tau'))$ is μ-infeasible, we have that $H(\mu)$ is not connected in Γ', given that the embedding of $pert(\tau')$ in Γ' is the same as in Γ, up to a flip, thus leading to a contradiction.
- If $\Gamma^1(pert(\tau'))$ is μ-kernelized, then either there exists a neighbor of τ in $\mathcal{T}$ different from τ' containing a vertex in μ, thus implying that $H(\mu)$ is not connected in Γ', given that the embedding of $pert(\tau')$ in Γ' is the same as in Γ, up to a flip, or there exists no neighbor of τ in $\mathcal{T}$ different from τ' containing a vertex in μ, thus implying that $H(\mu)$ is connected in Γ; in both cases this leads to a contradiction.
- If $\Gamma^1(pert(\tau'))$ is μ-bisided, then, in order for $H(\mu)$ to be connected in Γ', we have that $\Gamma^1(pert(\tau''))$ is μ-traversable for each neighbor τ'' of τ in $\mathcal{T}$ different from τ'. Since the embedding of $pert(\tau'')$ in Γ' is the same as in Γ, up to a flip, it follows that τ'' is μ-traversable in Γ, as well, thus $H(\mu)$ is connected in Γ, thus leading to a contradiction.

Hence, we can assume that, for each neighbor τ' of τ in $\mathcal{T}$, $\Gamma^1(pert(\tau'))$ (and hence $\Gamma^2(pert(\tau'))$) is either μ-sided, or μ-traversable, or it contains no vertex of μ. Consider a maximal sequence $\tau_1, \tau_2, \ldots, \tau_k$ of neighbors of τ in $\mathcal{T}$, ordered as in $\Gamma(sk(\tau))$, such that τ_i contains a vertex in μ. We distinguish two cases.

- If there exists no neighbor τ' of τ in $\mathcal{T}$ such that $\Gamma^1(pert(\tau'))$ is μ-traversable, then, by Property (C) of $\Gamma(sk(\tau))$, there exist at most two neighbors τ_1 and τ_2 of τ in $\mathcal{T}$ such that $\Gamma^1(pert(\tau_1))$ and $\Gamma^1(pert(\tau_2))$ are μ-sided, and moreover τ_1 and τ_2 are adjacent in $\Gamma(sk(\tau))$. By construction, the embedding of $pert(\tau_1)$ is chosen to be $\Gamma^1(pert(\tau_1))$ or $\Gamma^2(pert(\tau_1))$ so that a vertex in $pert(\tau_1)$ and in μ is incident to the face of $\Gamma(sk(\tau))$ to which τ_2 is incident. Analogously, the embedding of $pert(\tau_2)$ is chosen to be $\Gamma^1(pert(\tau_2))$ or $\Gamma^2(pert(\tau_2))$ so that a vertex in $pert(\tau_2)$ and in μ is incident to the face of $\Gamma(sk(\tau))$ to which τ_1 is incident. Hence, the subgraph of $H(\mu)$ in Γ induced by the vertices in τ_1 and the subgraph of $H(\mu)$ in Γ induced by the vertices in τ_2 are connected by an edge; moreover, both such subgraphs are connected. It follows that $H(\mu)$ is connected in Γ, a contradiction.
- If there exists at least one neighbor of τ in $\mathcal{T}$ whose embedding in Γ' is μ-traversable, then, by Properties (B) and (C) of $\Gamma(sk(\tau))$, the order in $\Gamma(sk(\tau))$ of the neighbors of τ in $\mathcal{T}$ is $\tau_1, \tau_2, \ldots, \tau_k$, where $\tau_2, \ldots, \tau_{k-1}$ are μ-traversable, where τ_1 and τ_k are μ-sided (and they might not exist), and where $k \geq 3$. By construction, the embedding of $pert(\tau_1)$, if τ_1 exists, is chosen to be

$\Gamma^1(pert(\tau_1))$ or $\Gamma^2(pert(\tau_1))$ so that a vertex in $pert(\tau_1)$ and in μ is incident to the face of $\Gamma(sk(\tau))$ to which τ_2 is incident. Analogously, the embedding of $pert(\tau_k)$, if τ_k exists, is chosen to be $\Gamma^1(pert(\tau_k))$ or $\Gamma^2(pert(\tau_k))$ so that a vertex in $pert(\tau_k)$ and in μ is incident to the face of $\Gamma(sk(\tau))$ to which τ_{k-1} is incident. Hence, the subgraph of $H(\mu)$ in Γ induced by the vertices in τ_1 is connected by an edge to the subgraph of $H(\mu)$ in Γ induced by the vertices in $\tau_2, \ldots, \tau_{k-1}$; moreover, the subgraph of $H(\mu)$ in Γ induced by the vertices in τ_k is connected by an edge to the subgraph of $H(\mu)$ in Γ induced by the vertices in $\tau_2, \ldots, \tau_{k-1}$; furthermore, these three subgraphs are connected. It follows that $H(\mu)$ is connected in Γ, a contradiction.

This concludes the proof of the claim. □

Claim 4.9 *Drawing Γ^C satisfies Condition (ii) of Lemma 4.1.*

Proof: We prove that Γ^C contains no cycle whose vertices all belong to the same cluster μ and whose interior contains a vertex v not in μ. We first prove this statement for drawing Γ, and we will later show how modifying the outer face of Γ to obtain Γ^C does not create a cycle whose vertices all belong to the same cluster μ and whose interior contains a vertex not in μ.

Suppose, for a contradiction, that Γ contains a cycle $\mathcal{C}$ whose vertices all belong to the same cluster μ and whose interior contains a vertex v not in μ. First, we discuss the existence of a cycle $\mathcal{C}$ violating Condition (ii) of Lemma 4.1 in the drawing constructed before.

If $\mathcal{C}$ entirely belongs to $pert(\tau')$, for some neighbor τ' of τ in $\mathcal{T}$, then $\mathcal{C}$ contains v in its interior also in Γ', given that the embedding of $pert(\tau')$ in Γ is the same as in Γ', up to a flip. However, by Lemma 4.1, this implies that Γ' is not a $\langle 0, 0, \infty \rangle$-drawing, a contradiction.

Otherwise, $\mathcal{C}$ is composed of two paths connecting the poles of τ, where the first path $P_\mu(\tau'')$ entirely belongs to $pert(\tau'')$ and the second path $P_\mu(\tau''')$ entirely belongs to $pert(\tau''')$, for two distinct neighbors τ'' and τ''' of τ in $\mathcal{T}$. Hence, $\Gamma^1(pert(\tau''))$ and $\Gamma^1(pert(\tau'''))$ (and hence $\Gamma^2(pert(\tau''))$ and $\Gamma^2(pert(\tau'''))$) are μ-spined. Moreover, they are both μ-side-spined (and possibly μ-full). Namely, if one of them, say $\Gamma^1(pert(\tau''))$, is μ-central-spined, then $\mathcal{C}$ contains a vertex not in μ in Γ', thus implying that Γ' does not satisfy Condition (ii) of Lemma 4.1 and hence that Γ' is not a $\langle 0, 0, \infty \rangle$-drawing, a contradiction.

Assume that $\Gamma^1(pert(\tau''))$ and $\Gamma^1(pert(\tau'''))$ (and, hence, $\Gamma^2(pert(\tau''))$ and $\Gamma^2(pert(\tau'''))$) are both μ-side-spined (possibly μ-full). By Property (A) of $\Gamma(sk(\tau))$, all the virtual edges, if any, that lie inside the cycle composed of the virtual edges

representing τ'' and τ''' in $\Gamma(sk(\tau))$ are μ-full. It follows that, if $\mathcal{C}$ contains a vertex not in μ in its interior, then such a vertex belongs to $pert(\tau'')$ or to $pert(\tau''')$. Suppose the former, the discussion for the latter case being analogous. Observe that this implies that $\Gamma^1(pert(\tau''))$ is not μ-full. Then, denote by $Q_\mu(\tau'')$ the path that is composed only of vertices of μ, that connects the poles of τ, and that delimits the outer face of $\Gamma^1(pert(\tau''))$.

Consider the two neighbors σ_1 and σ_2 of τ in $\mathcal{T}$ such that the virtual edges representing σ_1 and σ_2 are consecutive with the virtual edge representing τ'' in $\Gamma(sk(\tau))$. Assume, w.l.o.g., that the virtual edge representing σ_1 is internal to the cycle composed of the virtual edges representing τ'' and τ''' in $\Gamma(sk(\tau))$ (if any virtual edge internal to such a cycle exists) or that σ_1 coincides with τ''' (otherwise).

If σ_2 is not μ-spined, then by Rule RII-1, the embedding of $pert(\tau'')$ is chosen in such a way that $Q_\mu(\tau'')$ is incident to the face of $\Gamma(sk(\tau))$ the virtual edge representing σ_1 is incident to. However, this implies that $\mathcal{C}$ does not contain any vertex not belonging to μ in its interior, a contradiction.

Otherwise, σ_2 is μ-spined, and Rule RII-1 was applied to choose the embedding of $pert(\tau'')$ in such a way that $Q_\mu(\tau'')$ is incident to the face of $\Gamma(sk(\tau))$ the virtual edge representing σ_2 is incident to.

It follows that the virtual edges representing σ_2 and τ'' delimit the outer face of $\Gamma(sk(\tau))$. Namely, if that's not the case, then the cycle that is composed of the virtual edges representing σ_2 and τ''' would contain the virtual edge representing τ'', which is not μ-full, in its interior in $\Gamma(sk(\tau))$, thus contradicting the fact that $\Gamma(sk(\tau))$ satisfies Property (A). However, that the virtual edges representing σ_2 and τ'' delimit the outer face of $\Gamma(sk(\tau))$ contradicts the assumption that Rule RII-1 was applied to choose the embedding of $pert(\tau'')$ in such a way that $Q_\mu(\tau'')$ is incident to the face of $\Gamma(sk(\tau))$ the virtual edge representing σ_2 is incident to, thus obtaining a contradiction.

This completes the proof that Γ does not contain a cycle $\mathcal{C}$ whose vertices all belong to the same cluster μ and whose interior contains a vertex v not in μ.

Now we deal with Γ^C. If Γ^C has the same outer face as Γ, then there is nothing to prove. Otherwise, suppose, for a contradiction, that Γ^C contains a cycle $\mathcal{C}$ whose vertices all belong to the same cluster μ and whose interior contains a vertex v not in μ. Since Γ and Γ^C coincide when restricted to the pertinent graphs of the children of τ in $\mathcal{T}$, it follows that $\mathcal{C}$ is composed of a path $P_\mu(\sigma)$ in the pertinent graph of the parent σ of τ in $\mathcal{T}$ and of a path $P_\mu(\sigma')$ in the pertinent graph of a neighbor $\sigma' \neq \sigma$ of τ in $\mathcal{T}$. Also, since rule R0 sets the embedding of $pert(\sigma)$ so that e_ρ is incident to the outer face, it follows that either rule RI-1, or rule RI-2, or rule RII-1 was applied to determine the embedding of $pert(\sigma)$ to be either $\Gamma^1(pert(\sigma))$ or $\Gamma^2(pert(\sigma))$, so that e_ρ is not incident to the outer face of Γ. We distinguish two cases based on which rule was applied to determine the embedding of $pert(\sigma)$.

Suppose first that rule RII-1 was applied to determine the embedding of $pert(\sigma)$. By the assumptions of rule RII-1, $\Gamma^1(pert(\sigma))$ is ν-side-spined and not ν-full, for some cluster $\nu \in T$ (possibly $\nu = \mu$), and the embedding of the pertinent graph of a neighbor σ' of σ in $\Gamma(sk(\tau))$ is ν-spined. Also, since e_ρ is not incident to the outer face of Γ, it follows that the path $P_\nu(\sigma)$ in $pert(\sigma)$ that connects the poles of σ, that delimits the outer face of $\Gamma^1(pert(\sigma))$, and that entirely belongs to ν contains e_ρ. However, since the embedding of $pert(\sigma)$ in Γ^C coincides with the embedding of $pert(\sigma)$ in Γ' and since e_ρ is incident to the outer face of Γ', it follows that the cycle $\mathcal{C}'$ composed of $P_\nu(\sigma)$ and of any path in $pert(\sigma')$ that connects the poles of σ' and that entirely belongs to ν (such a path exists because σ' is ν-spined) contains in its interior in Γ' a vertex not belonging to ν (namely any vertex in $pert(\sigma)$ not in ν; such a vertex exists because $pert(\sigma)$ is not ν-full). Thus, Γ' does not satisfy Condition (ii) of Lemma 4.1, a contradiction.

Suppose next that rule RI-1 or rule RI-2 was applied to determine the embedding of $pert(\sigma)$. By the assumptions of rules RI-1 and RI-2, $\Gamma^1(pert(\sigma))$ is ν-sided, for some cluster $\nu \in T$, and the embedding of the pertinent graph of a neighbor σ' of σ in $\Gamma(sk(\tau))$ is ν-traversable or ν-sided.

- If $\Gamma^1(pert(\sigma))$ is μ-central-spined, then Γ and Γ' contain cycles that entirely belong to μ and that contain in their interior vertices not belonging to μ (namely vertices in $pert(\sigma)$), thus contradicting the fact that Γ and Γ' are $\langle 0, 0, \infty \rangle$-drawings of $C(G, T)$.
- If $\Gamma^1(pert(\sigma))$ is μ-side-spined and the path that connects the poles of σ, that delimits the outer face of $\Gamma^1(pert(\sigma))$, and that entirely belongs to μ contains e_ρ, then Γ' contains a cycle that entirely belongs to μ and that contains in its interior a vertex not belonging to μ (namely a vertex in $pert(\sigma)$), thus contradicting the fact that Γ' is a $\langle 0, 0, \infty \rangle$-drawing of $C(G, T)$.
- If $\Gamma^1(pert(\sigma))$ is μ-side-spined and the path that connects the poles of σ, that delimits the outer face of $\Gamma^1(pert(\sigma))$, and that entirely belongs to μ does not contain e_ρ, then Γ contains a cycle that entirely belongs to μ and that contains in its interior a vertex not belonging to μ (namely a vertex in $pert(\sigma)$), thus contradicting the fact that Γ is a $\langle 0, 0, \infty \rangle$-drawing of $C(G, T)$.

We can hence assume that $pert(\sigma)$ is μ-full. Denote by $\sigma_1, \sigma_2, \ldots, \sigma_l$ the order of the children of τ as in $\Gamma(sk(\tau))$, where σ_1 is incident to the outer face of $\Gamma(sk(\tau))$ in Γ^C, while σ_l is incident to the outer face of $\Gamma(sk(\tau))$ in Γ.

If any μ-full child σ_j of τ in $\mathcal{T}$ exists, then all of $\sigma_1, \sigma_2, \ldots, \sigma_j$ are μ-full, as otherwise Γ would not satisfy Condition (ii) of Lemma 4.1. Denote by f the largest

index such that $\sigma_1, \sigma_2, \ldots, \sigma_f$ are μ-full. Also, if any ν-traversable child σ_j of τ in $\mathcal{T}$ exists, then all of $\sigma_1, \sigma_2, \ldots, \sigma_j$ are ν-traversable, as otherwise Γ would not satisfy Condition (i) of Lemma 4.1. Denote by t the largest index such that $\sigma_1, \sigma_2, \ldots, \sigma_t$ are ν-traversable.

- If $t \geq f > 0$, then every μ-full child of τ is ν-traversable. All the μ-full children of τ occur consecutively in $\Gamma'(sk(\tau))$, as otherwise Γ' does not satisfy Condition (ii) of Lemma 4.1. Since $f > 0$, at least one μ-full and ν-traversable child of τ exists, and it is next to σ in $\Gamma'(sk(\tau))$. Hence, either a child of τ that is not ν-traversable exists, thus implying that $H(\nu)$ is not connected in Γ' and hence that Γ' does not satisfy Condition (i) of Lemma 4.1, a contradiction, or every child of τ is ν-traversable, thus implying that rules RI-1 and RI-2 were not applied to determine the embedding of $pert(\sigma)$, a contradiction.

- If $f \geq t > 0$, then every ν-traversable child of τ is μ-full. All the μ-full children of τ occur consecutively in $\Gamma'(sk(\tau))$, as otherwise Γ' does not satisfy Condition (ii) of Lemma 4.1. Since $t > 0$, at least one μ-full and ν-traversable child of τ exists. Hence, either a child of τ exists that is not ν-traversable and not μ-full exists, thus implying that $H(\nu)$ is not connected in Γ' and hence that Γ' does not satisfy Condition (i) of Lemma 4.1, or every child of τ is μ-full, thus contradicting the assumption that $\mathcal{C}$ contains a vertex not in μ in its interior.

- If $t = 0$, then σ_1 is ν-sided, as otherwise neither rule RI-1 nor rule RII-2 would be applied to determine the embedding of $pert(\sigma)$. It follows that σ_1 is incident to the outer face of $\Gamma'(sk(\tau))$. Thus, if σ_1 is μ-spined, then Γ' does not satisfy Condition (ii) of Lemma 4.1, a contradiction, while if σ_1 is not μ-spined (and some other child of τ is), then Γ does not satisfy Condition (ii) of Lemma 4.1, a contradiction.

- If $f = 0$, then there exists exactly one child of τ that is μ-spined. If σ_1 is not μ-spined, then Γ does not satisfy Condition (ii) of Lemma 4.1, a contradiction. Otherwise σ_1 is μ-spined. Hence, σ_1 is next to σ in $\Gamma'(sk(\tau))$. Thus, if σ_1 is ν-traversable, then either every child of τ is ν-traversable, thus implying that rules RI-1 and RI-2 were not applied to determine the embedding of $pert(\sigma)$, a contradiction, or a child of τ that is not ν-traversable exists, thus implying that $H(\nu)$ is not connected in Γ' and hence that Γ' does not satisfy Condition (i) of Lemma 4.1, a contradiction. Finally, if σ_1 is ν-sided, then $H(\nu)$ is not connected either in Γ or in Γ', a contradiction.

This concludes the proof of the claim. □

Claims 4.4–4.6 prove that Properties (A), (B), and (C) of Lemma 4.5 are necessary in order for an embedding $\Gamma(sk(\tau))$ in which the edge e representing the parent of τ is incident to the outer face to be extensible. The sufficiency of Properties (A), (B), and (C) of Lemma 4.5 is easily proved as described above if τ is an S-node or an R-node; Claims 4.7–4.9 prove the sufficiency of Properties (A), (B), and (C) of Lemma 4.5 if τ is a P-node. This completes the proof of Lemma 4.5.

We are now ready to give an algorithm for testing whether a given clustered graph admits a $\langle 0, 0, \infty \rangle$-drawing.

Theorem 4.4 *Let $C(G, T)$ be a clustered graph such that G is biconnected. There exists a polynomial-time algorithm to test whether $C(G, T)$ admits a $\langle 0, 0, \infty \rangle$-drawing.*

Proof: Let $\mathcal{T}$ be the SPQR-tree of G. Consider any Q-node ρ of $\mathcal{T}$ and root $\mathcal{T}$ at ρ. Such a choice corresponds to assuming that any considered planar embedding of G has the edge e_ρ of G corresponding to ρ incident to the outer face. In the following, we describe how to test whether $C(G, T)$ admits a $\langle 0, 0, \infty \rangle$-drawing under the above assumption. The repetition of such a test for all possible choices of ρ results in a test of whether $C(G, T)$ admits a $\langle 0, 0, \infty \rangle$-drawing.

First, we perform a preprocessing step to compute the following information.

For each node $\tau \in \mathcal{T}$ and for each cluster $\mu \in T$, we label τ (and the virtual edge representing τ in the skeleton of the parent of τ) with flags stating whether τ is (i) μ-touched, (ii) τ is μ-full, and (iii) τ is μ-spined.

Observe that such information can be easily computed in polynomial time based on whether any vertex of $pert(\tau)$ different from its poles belongs to μ, on whether all the vertices of $pert(\tau)$ belong to μ, and on whether there exists a path in $pert(\tau)$ connecting the poles of τ and entirely belonging to μ, respectively. In particular, such information does not change if the embedding of $pert(\tau)$ varies.

Also, for each node $\tau \in \mathcal{T}$ and for each cluster $\mu \in T$, we label τ (and the virtual edge representing τ in the skeleton of the parent of τ) with a flag stating whether τ is μ-traversable in any planar embedding that is not μ-infeasible. Namely, by Lemma 4.3, if an embedding of $pert(\tau)$ is μ-traversable, then every embedding of $pert(\tau)$ that is not μ-infeasible is μ-traversable. To compute this information, we traverse $\mathcal{T}$ bottom-up while computing, for each encountered node τ' of $\mathcal{T}$, whether $H(\tau', \mu)$ has a connected component that contains $f'(\tau')$ and $f''(\tau')$ in any planar embedding of $pert(\tau')$. In particular:

- If τ' is a Q-node, then we label τ' as μ-traversable if and only if one of the poles of τ' belongs to μ;

- if τ' is an S-node, then we label τ' as μ-traversable if and only if at least one of the children of τ' is labeled as μ-traversable;
- if τ' is a P-node, then we label τ' as μ-traversable if and only if every child of τ' is labeled as μ-traversable;
- if τ' is an R-node, then we label τ' as μ-traversable if and only if, in the unique (up to a flip) planar embedding $\Gamma(sk(\tau'))$ of $sk(\tau')$, there exists a sequence $(\tau_1, \tau_2, \ldots, \tau_x)$ of children of τ such that: (1) τ_j is a child of τ that is labeled as μ-traversable, for every $1 \leq j \leq x$, (2) the virtual edges representing τ_1 and τ_x are incident to the two faces to which the virtual edge representing the parent of τ is incident to, and (3) the virtual edges representing τ_j and τ_{j+1} are incident to a common face in $\Gamma(sk(\tau'))$, for every $1 \leq j \leq x-1$.

Observe that, whether $pert(\tau)$ is μ-sided, μ-bisided, μ-kernelized, μ-infeasible, μ-side-spined, or μ-central-spined depends on the actual embedding of $pert(\tau)$.

Second, we traverse $\mathcal{T}$ bottom-up. For every P-node and every R-node τ of $\mathcal{T}$, the *visible nodes* of τ are the children of τ that are not S-nodes plus the children of each S-node that is a child of τ. At each step, we consider either a P-node or an R-node τ with visible nodes $\tau_1, \ldots, \tau_k$. We inductively assume that, for each visible node τ_i, with $1 \leq i \leq k$, an extensible embedding $\Gamma(pert(\tau_i))$ has been computed, together with the information whether $\Gamma(pert(\tau_i))$ is μ-traversable, μ-sided, μ-bisided, μ-kernelized, μ-infeasible, μ-side-spined, or μ-central-spined, for each cluster μ in T.

We show how to test whether an extensible embedding $\Gamma(pert(\tau))$ of $pert(\tau)$ exists. Such a test consists of two phases. We first test whether $sk(\tau)$ admits an extensible embedding $\Gamma(sk(\tau))$. In the negative case, we can conclude that $C(G,T)$ has no $\langle 0, 0, \infty \rangle$-drawing in which e_ρ is incident to the outer face. In the positive case, we also test whether a flip of each $\Gamma(pert(\tau_i))$ exists such that the resulting embedding $\Gamma(pert(\tau))$ is extensible.

Extensible embedding of the skeleton of τ.

Suppose that τ is an R-node. Then, $sk(\tau)$ has a unique embedding, up to a flip. We hence test whether Properties (A), (B), and (C) of Lemma 4.5 are satisfied. Observe that such a test can be easily performed in polynomial time, based on the information on whether the visible nodes of μ (and hence the children of τ) are μ-spined, μ-full, μ-touched, and μ-traversable.

Suppose that τ is a P-node. We check whether there exists an extensible embedding $\Gamma(sk(\tau))$ of $sk(\tau)$ as follows. We impose constraints on the ordering of the virtual edges of τ.

A first set of constraints establishes that $\Gamma(sk(\tau))$ satisfies Property (A) of Lemma 4.5. Namely, for each cluster μ:

(a) We constrain all the μ-full virtual edges to be consecutive;

(b) if there exists no μ-full virtual edge, then we constrain each pair of μ-spined virtual edges to be consecutive; and

(c) if there exists at least one μ-full virtual edge, then, for each μ-spined virtual edge, we constrain such an edge and all the μ-full virtual edges to be consecutive.

A second set of constraints establishes that $\Gamma(sk(\tau))$ satisfies Properties (B) and (C) of Lemma 4.5. Namely, for each cluster μ:

(a) We constrain all the μ-traversable virtual edges to be consecutive;

(b) if there exists no μ-traversable virtual edge, then we constrain each pair of μ-touched virtual edges to be consecutive; and

(c) if there exists at least one μ-traversable virtual edge, then, for each μ-touched virtual edge, we constrain such an edge and all the μ-full virtual edges to be consecutive.

We check whether an ordering of the virtual edges of $sk(\tau)$ that enforces all these constraints exists by using the PQ-tree data structure [BL76]. If such an ordering does not exist, we conclude that $sk(\tau)$ admits no extensible embedding. Otherwise, we have an embedding $\Gamma(sk(\tau))$ of $sk(\tau)$ which satisfies Properties (A)–(C), hence it is extensible.

Extensible embedding of the pertinent graph of τ.

We now determine, for each node τ of $\mathcal{T}$ that is either an R- or a P-node, an extensible embedding $\Gamma(pert(\tau))$ of $pert(\tau)$, if one exists. This is done by choosing the flip of the embedding $\Gamma(pert(\tau_i))$ of each visible node τ_i of τ in such a way that $\Gamma(pert(\tau))$ satisfy Properties (i) and (ii) of Lemma 4.1. Observe that the choice of the flip of the embedding $\Gamma(pert(\tau_i))$ of each visible node τ_i of τ, together with the choice of the embedding of $sk(\tau)$ to be $\Gamma(sk(\tau))$, completely determines $\Gamma(pert(\tau))$.

We will construct a 2-SAT formula F such that $pert(\tau)$ admits an extensible embedding if and only if F is satisfiable. We initialize $F = \emptyset$. Then, for each visible node τ_i of τ, we assign an arbitrary flip to τ_i and define a boolean variable x_i that is positive if τ_i has the assigned flip and negative otherwise.

We first add some clauses to F in order to ensure that $\Gamma(pert(\tau))$ satisfies Property (ii) of Lemma 4.1.

For each cluster μ, we consider the embedded subgraph $\Gamma_\mu(sk(\tau))$ of $\Gamma(sk(\tau))$ containing all the μ-spined virtual edges. Note that, since $\Gamma(sk(\tau))$ satisfies Property

(A) of Lemma 4.5, each edge of $\Gamma_\mu(sk(\tau))$ that is not incident to the outer face of $\Gamma_\mu(sk(\tau))$ is μ-full. Hence, no flip choice has to be done for these edges.

Consider any edge g in $\Gamma_\mu(sk(\tau))$ such that g is incident to the external face and to an internal face f_g of $\Gamma_\mu(sk(\tau))$. Consider each visible node τ_i of τ such that: (i) either g corresponds to τ_i or g corresponds to the S-node which is the parent of τ_i and (ii) $\Gamma(pert(\tau_i))$ is either μ-side-spined or μ-central-spined.

Then, if $\Gamma(pert(\tau_i))$ is μ-central-spined, then we conclude that $pert(\tau)$ has no extensible embedding (with e incident to the outer face). Otherwise, $\Gamma(pert(\tau_i))$ is μ-side-spined. In this case, add clause $\{x_i\}$ to F if the default flip of $\Gamma(pert(\tau_i))$ does not place any vertex not in μ on f_g, and add clause $\{\neg x_i\}$ to F if the default flip of $\Gamma(pert(\tau_i))$ places a vertex not in μ on f_g.

We next add some clauses to F in order to ensure that $\Gamma(pert(\tau))$ satisfy Property (i) of Lemma 4.1.

Suppose that there exists a visible node τ_i of τ such that: (i) $\Gamma(pert(\tau_i))$ is μ-bisided; and (ii) if τ_i is child of an S-node σ, then no child of σ is μ-traversable.

Then, we check:

(a) Whether all the visible nodes of τ that are children of τ, except for τ_i, are μ-traversable; and

(b) whether, for each S-node γ that is child of τ and that is not the parent of τ_i, at least one child of γ is μ-traversable.

If one of the checks fails, we conclude that $pert(\tau)$ has no extensible embedding (with e incident to the outer face), otherwise we do not add any clause to F and we continue as follows.

Suppose that there exists a visible node τ_i of τ such that:

(i) $\Gamma(pert(\tau_i))$ is μ-sided; (ii) if τ_i is child of an S-node σ, then no child of σ is μ-traversable; and (iii) τ_i shares exactly one face with a μ-traversable or μ-sided node τ_j.

Then, add clause $\{x_i\}$ to F if the default flip of $\Gamma(pert(\tau_i))$ places the vertices of $pert(\tau_i)$ belonging to μ on the face that τ_i shares with τ_j, and add clause $\{\neg x_i\}$ to F otherwise.

Suppose that there exists an S-node σ child of τ such that:

(i) no child of σ is μ-traversable; and (ii) no child of τ different from σ is μ-touched.

Consider each pair of visible nodes τ_i and τ_j children of σ that are both μ-sided.

Then, add clauses $(x_i \vee \neg x_j)$ and $(\neg x_i \vee x_j)$ to F if the default flips of $\Gamma(pert(\tau_i))$ and of $\Gamma(pert(\tau_j))$ place vertices of $pert(\tau_i)$ and vertices of $pert(\tau_j)$ belonging to μ on the same face. Otherwise, add clauses $(x_i \vee x_j)$ and $(\neg x_i \vee \neg x_j)$ to F.

Observe that all the described checks and embedding choices, and the construction and solution of the 2-SAT formula can be easily performed in polynomial time. Once an embedding $\Gamma(pert(\tau))$ of $pert(\tau)$ has been computed, by traversing $\Gamma(pert(\tau))$ it can be determined in polynomial time whether such an embedding is μ-sided, μ-bisided, μ-kernelized, μ-side-spined, or μ-central-spined. By Lemma 4.4, if a $\langle 0, 0, \infty\rangle$-drawing of $C(G, T)$ exists, then there exists a $\langle 0, 0, \infty\rangle$-drawing of $C(G, T)$ in which the embedding of $pert(\tau)$ is $\Gamma(pert(\tau))$ or its flip. This allows the bottom-up visit of $\mathcal{T}$ to go through. The correctness of the embedding choices performed in order to construct $\Gamma(pert(\tau))$ follows from Lemmata 4.1, 4.4, and 4.5. This concludes the proof of the theorem. □

We now turn our attention to establish bounds on the minimum value of γ in a $\langle 0, 0, \gamma\rangle$-drawing of a clustered graph.

Theorem 4.5 *Let $C(G, T)$ be a clustered graph. There exists an algorithm to compute a $\langle 0, 0, \gamma\rangle$-drawing of $C(G, T)$ with $\gamma \in O(n^3)$, if any such drawing exists. If $C(G, T)$ is flat, then $\gamma \in O(n^2)$.*

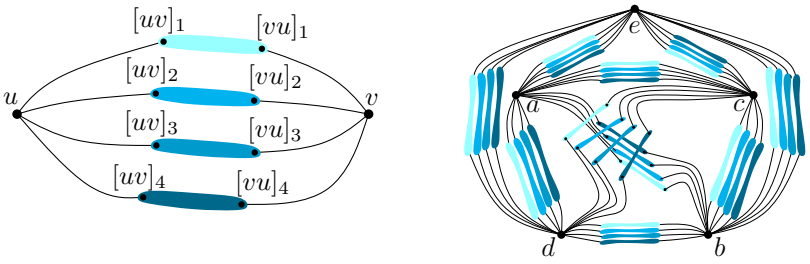
Proof: Suppose that $C(G, T)$ admits a $\langle 0, 0, \infty\rangle$-drawing. Then, consider the drawing Γ of the underlying graph G in any such a drawing. For each cluster μ, place a vertex $u_{\mu,f}$ inside any face f of Γ that contains at least one vertex belonging to μ, and connect $u_{\mu,f}$ to all the vertices of μ incident to f. Note that, the graph composed by the vertices of μ and by the added vertices u_{μ,f_i} is connected. Then, construct a spanning tree of such a graph and draw $R(\mu)$ slightly surrounding such a spanning tree. The cubic bound on γ comes from the fact that each of the $O(n)$ clusters crosses each of the $O(n)$ other clusters a linear number of times. On the other hand, if $C(G, T)$ is flat, then each of the $O(n)$ clusters crosses each of the $O(n)$ other clusters just once. □

4.4 Lower bounds

In this section we give lower bounds on the number of ee-, er-, and rr-crossings in $\langle \alpha, \beta, \gamma\rangle$-drawings of clustered graphs.

First, we prove an auxiliary lemma concerning the crossing number in graphs without a cluster hierarchy. Given a graph G, we define $G(m)$ as the multigraph obtained by replacing each edge of G with a set of m multiple edges. For each pair (u, v) of vertices, we denote by $S(u, v)$ the set of multiple edges connecting u and v.

Lemma 4.6 *Graph $G(m)$ has crossing number $cr(G(m)) \geq m^2 \cdot cr(G)$.*



(a) (b)

Figure 4.16: Illustrations for the proof of Theorem 4.6. (a) Edges and clusters in $M(u, v)$. (b) Clustered graph $C(G, T)$.

Proof: Consider a drawing Γ of $G(m)$ with the minimum number $cr(G(m))$ of crossings.

First, observe that in Γ no edge intersects itself, no two edges between the same pair of vertices intersect, and each pair of edges crosses at most once. Namely, if any of these conditions does not hold, it is easy to modify Γ to obtain another drawing of $G(m)$ with a smaller number of crossings, which is not possible by hypothesis (see, e.g., [PT00a]).

We show that there exists a drawing Γ' of $G(m)$ with $cr(G(m))$ crossings in which, for each pair of vertices u and v, all the edges between u and v cross the same set of edges in the same order. Let $e_{min}(u, v)$ be any edge with the minimum number of crossings among the edges of $S(u, v)$. Redraw all the edges in $S(u, v) \setminus e_{min}(u, v)$ so that they intersect the same set of edges as $e_{min}(u, v)$, in the same order as $e_{min}(u, v)$. Repeating this operation for each set $S(u, v)$ yields a drawing Γ' with the required property.

Starting from Γ', we construct a drawing Γ_G of G. For each set of edges $S(u, v)$ remove all edges except for one edge $e^*(u, v)$. The resulting drawing Γ_G of G has at least $cr(G)$ crossings, by definition. For any two edges $e^*(u, v)$ and $e^*(w, z)$ that cross in Γ_G, we have that each edge in $S(u, v)$ crosses each edge in $S(w, z)$, by the properties of Γ'. Hence Γ' contains at least $m^2 \cdot cr(G)$ crossings. □

We prove a lower bound on the total number of crossings in an $\langle \alpha, \beta, \gamma \rangle$-drawing of a clustered graph when all the three types of crossings are admitted.

Theorem 4.6 *There exists an n-vertex non-c-connected flat clustered graph $C(G,T)$ that admits $\langle\infty,0,0\rangle$-, $\langle 0,\infty,0\rangle$-, and $\langle 0,0,\infty\rangle$-drawings, and such that $\alpha+\beta+\gamma \in \Omega(n^2)$ in every $\langle\alpha,\beta,\gamma\rangle$-drawing of $C(G,T)$.*

Proof: Clustered graph $C(G,T)$ is as follows. Initialize graph G with five vertices a,b,c,d,e. For each two vertices $u,v \in \{a,b,c,d,e\}$, with $u \neq v$, and for $i = 1,\ldots,m$, add to G vertices $[uv]_i, [vu]_i$, and edges $(u,[uv]_i)$ and $(v,[vu]_i)$, and add to T a cluster $\mu(u,v)_i = \{[uv]_i,[vu]_i\}$. Vertices a,b,c,d,e belong to clusters $\mu_a,\mu_b,\mu_c,\mu_d,\mu_e$, respectively. See Fig. 4.16. We denote by $M(u,v) = \{(u,[uv]_i),(v,[vu]_i),\mu(u,v)_i | i = 1,\ldots,m\}$.

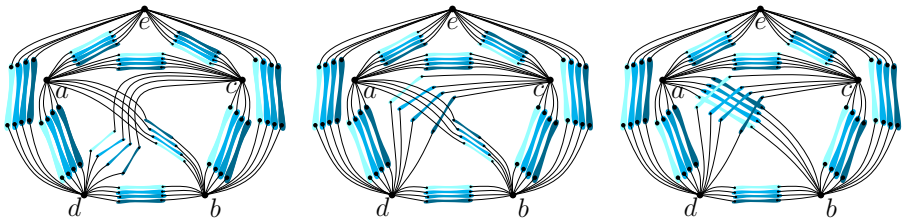
First, we prove that C admits $\langle\infty,0,0\rangle$-, $\langle 0,\infty,0\rangle$-, and $\langle 0,0,\infty\rangle$-drawings. Consider a drawing Γ^* of K_5 on vertices $\{a,b,c,d,e\}$ with one crossing. Assume, without loss of generality, that the crossing is on (a,b) and (c,d). For each pair of vertices $u,v \in \{a,b,c,d,e\}$, with $(u,v) \notin \{(a,b),(c,d)\}$, replace edge (u,v) in Γ^* with $M(u,v)$ in such a way that the drawing of the edges and clusters in $M(u,v)$ is arbitrarily close to the drawing of (u,v). It remains to draw edges and clusters in $M(a,b)$ and $M(c,d)$. This is done differently for $\langle\infty,0,0\rangle$-, $\langle 0,\infty,0\rangle$-, and $\langle 0,0,\infty\rangle$-drawings.

$\langle\infty,0,0\rangle$-drawing Replace (a,b) and (c,d) in Γ^* with $M(a,b)$ and $M(c,d)$ in such a way that the drawing of the edges and clusters in $M(a,b)$ (in $M(c,d)$) is arbitrarily close to the drawing of (a,b) (of (c,d)) and for each $1 \leq i,j \leq m$, edge $(a,[ab]_i)$ crosses edge $(c,[cd]_j)$, while edges $(b,[ba]_i)$ and $(d,[dc]_i)$, and regions $R(\mu(a,b)_i)$ and $R(\mu(c,d)_j)$ are not involved in any crossing. See Fig. 4.17(a).

$\langle 0,\infty,0\rangle$-drawing Replace (a,b) and (c,d) in Γ^* with $M(a,b)$ and $M(c,d)$ in such a way that the drawing of the edges and clusters in $M(a,b)$ (in $M(c,d)$) is arbitrarily close to the drawing of (a,b) (of (c,d)) and for each $1 \leq i,j \leq m$, edge $(a,[ab]_i)$ crosses region $R(\mu(c,d)_j)$, while edges $(b,[ba]_i)$, $(c,[cd]_i)$, and $(d,[dc]_j)$, and region $R(\mu(a,b)_i)$ are not involved in any crossing. See Fig. 4.17(b).

$\langle 0,0,\infty\rangle$-drawing Replace (a,b) and (c,d) in Γ^* with $M(a,b)$ and $M(c,d)$ in such a way that the drawing of the edges and clusters in $M(a,b)$ (in $M(c,d)$) is arbitrarily close to the drawing of (a,b) (of (c,d)) and for each $1 \leq i,j \leq m$, region $R(\mu(a,b)_i)$ crosses region $R(\mu(c,d)_j)$, while edges $(a,[ab]_i)$, $(b,[ba]_i)$, $(c,[cd]_j)$, and $(d,[dc]_i)$ are not involved in any crossing. See Fig. 4.17(c).

Second, we show that $\alpha+\beta+\gamma \in \Omega(n^2)$ in every $\langle\alpha,\beta,\gamma\rangle$-drawing of $C(G,T)$. Consider any such a drawing Γ. Starting from Γ, we obtain a drawing Γ' of a subdivision of $K_5(m)$ as follows. For each $u,v \in \{a,b,c,d,e\}$, with $u \neq v$, and for



(a) (b) (c)

Figure 4.17: (a) $\langle \infty, 0, 0 \rangle$-drawing of C. (b) $\langle 0, \infty, 0 \rangle$-drawing of C. (c) $\langle 0, 0, \infty \rangle$-drawing of C.

each $i = 1, \ldots, m$, insert a drawing of edge $([uv]_i, [vu]_i)$ inside $R(\mu(u, v)_i)$ and remove region $R(\mu(u, v)_i)$. Further, remove regions $R(\mu_a)$, $R(\mu_b)$, $R(\mu_c)$, $R(\mu_d)$, and $R(\mu_e)$. The obtained graph is a subdivision of $K_5(m)$. Hence, by Lemma 4.6, Γ' has $\Omega(n^2)$ crossings. Moreover, each crossing in Γ' corresponds either to an edge-edge crossing, or to an edge-region crossing, or to a region-region crossing in Γ, thus proving the theorem. □

We now turn our attention to drawings in which only one type of crossings is allowed. In this setting, we show that the majority of the upper bounds presented in the previous section are tight by giving lower bounds on the number of crossings of $\langle \infty, 0, 0 \rangle$-, $\langle 0, \infty, 0 \rangle$-, and $\langle 0, 0, \infty \rangle$-drawings. As a corollary of Theorem 4.6, there exists a clustered graph $C(G, T)$ such that $\alpha \in \Omega(n^2)$ in every $\langle \alpha, 0, 0 \rangle$-drawing of $C(G, T)$, such that $\beta \in \Omega(n^2)$ in every $\langle 0, \beta, 0 \rangle$-drawing of $C(G, T)$, and such that $\gamma \in \Omega(n^2)$ in every $\langle 0, 0, \gamma \rangle$-drawing of $C(G, T)$. However, in the following we present quadratic lower bounds on restricted classes of clustered graphs and a cubic lower bound for $\langle 0, 0, \infty \rangle$-drawings of clustered graphs.

We first consider $\langle \infty, 0, 0 \rangle$-drawings. We give two lower bounds, which deal with c-connected and non-c-connected clustered graphs, respectively.

Theorem 4.7 *There exists a c-connected flat clustered graph $C(G, T)$ such that $\alpha \in \Omega(n^2)$ in every $\langle \alpha, 0, 0 \rangle$-drawing of $C(G, T)$.*

Proof: We first describe $C(G, T)$. Graph G is a subdivision of $K_5(m)$, with $m = \frac{n-5}{9}$, where the set of edges $S(d, e)$ has been removed. Tree T is such that $\mu_2 = \{d\}$, $\mu_3 = \{e\}$, and all the other vertices belong to μ_1. See Fig. 4.18(a). Since, in any $\langle \infty, 0, 0 \rangle$-drawing Γ of $C(G, T)$, both d and e must be outside any cycle composed

of vertices of μ_1 (as otherwise they would lie inside $R(\mu_1)$, see Fig. 4.18(b)), a set of m length-2 paths can be drawn in Γ between d and e without creating other crossings, thus obtaining a drawing of a subdivision of $K_5(m)$ in which the crossings are the same as in Γ (see Fig. 4.18(c)). Since $cr(K_5) = 1$ and since $m = \Omega(n)$, by Lemma 4.6, $\alpha \in \Omega(n^2)$. □

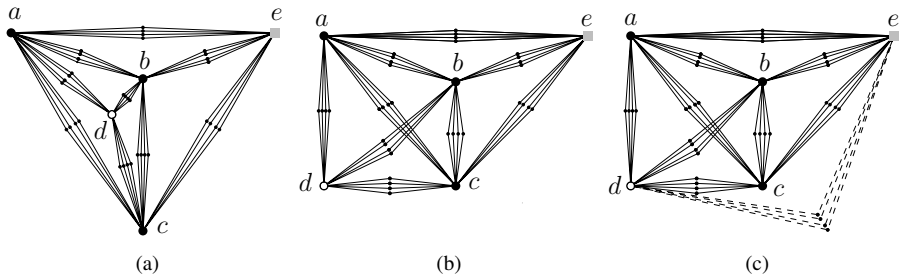


Figure 4.18: Illustration for the proof of Theorem 4.7. Vertices in μ_1 are black, vertices in μ_2 are white, and vertices in μ_3 are gray. (a) Graph G. (b) Vertices d and e must be outside all the cycles composed of vertices of μ_1. (c) Graph G'', where the length-2 paths connecting d and e are dashed.

Theorem 4.8 *There exists a non-c-connected flat clustered graph $C(G,T)$, where G is a matching, such that $\alpha \in \Omega(n^2)$ in every $\langle \alpha, 0, 0 \rangle$-drawing of $C(G,T)$.*

Proof: Clustered graph $C(G,T)$ is constructed as follows. Tree T is a star graph with five leaves $\mu_1, \ldots, \mu_5$. For each $i \neq j$ with $1 \leq i, j \leq 5$, add $\frac{n}{20}$ vertices to μ_i and to μ_j and construct a matching between these two sets of vertices.

Consider any $\langle \alpha, 0, 0 \rangle$-drawing Γ of $C(G,T)$ such that α is minimum. We prove that Γ does not contain any edge-edge crossing inside the regions representing clusters. This implies that, contracting the regions to single points yields a drawing of a subdivision of $K_5(n/20)$, and Lemma 4.6 applies to obtain the claimed lower bound for α.

Assume, for a contradiction, that a crossing between two edges e_1 and e_2 occurs inside the region $R(\mu)$ representing a cluster μ. Since Γ has no edge-region crossings, both e_1 and e_2 connect a vertex in μ with a vertex not in μ. Then, one might place the endvertex of e_1 belonging to μ arbitrarily close to the boundary of $R(\mu)$ in such a way that it does not cross e_2 inside $R(\mu)$. Since this operation reduces the number of crossings, we have a contradiction to the fact that α is minimum.

Then, we add a vertex to each cluster μ_i and connect it to all the vertices of μ_i. Observe that, since no two edges cross inside the region representing a cluster, such vertices and edges can be added without creating any new crossings.

Finally, removing from Γ the drawings of the regions representing the clusters leads to a drawing of a subdivision of $K_5(n/20)$ with α crossings. By Lemma 4.6, $\alpha \in \Omega(n^2)$. □

We now prove some lower bounds on the number of er-crossings in $\langle 0, \infty, 0\rangle$-drawings of clustered graphs. In the case of non-c-connected flat clustered graphs, a quadratic lower bound directly follows from Theorem 4.6, as stated in the following.

Corollary 4.1 *There exists a non-c-connected flat clustered graph $C(G,T)$ such that $\beta \in \Omega(n^2)$ in every $\langle 0, \beta, 0\rangle$-drawing of $C(G,T)$.*

Next, we deal with the c-connected case and present a quadratic and a linear lower bound for non-flat and flat cluster hierarchies, respectively.

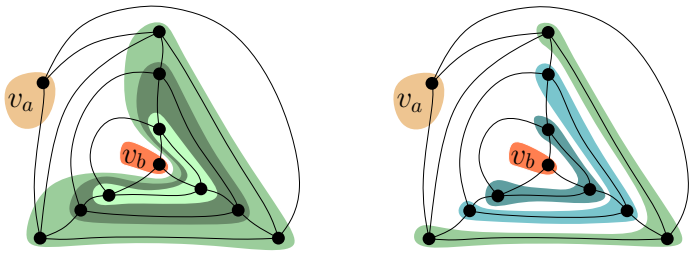
Theorem 4.9 *There exists a c-connected non-flat clustered graph $C(G,T)$ such that $\beta \in \Omega(n^2)$ in every $\langle 0, \beta, 0\rangle$-drawing of $C(G,T)$.*

Proof: Let G be an $(n+2)$-vertex triconnected planar graph such that for $i = 1, \ldots, \frac{n}{3}$, G contains a 3-cycle $C_i = (a_i, b_i, c_i)$. Further, for $i = 1, \ldots, \frac{n}{3} - 1$, G has edges (a_i, a_{i+1}), (b_i, b_{i+1}), (c_i, c_{i+1}). Finally, G contains two vertices v_a and v_b such that v_a is connected to a_1, b_1, c_1 and v_b is connected to $a_{\frac{n}{3}}, b_{\frac{n}{3}}, c_{\frac{n}{3}}$. Tree T is defined as follows: $\mu_1 = \{a_1, b_1, c_1\}$ and, for each $i = 2, \ldots, \frac{n}{3}$, $\mu_i = \mu_{i-1} \cup \{a_i, b_i, c_i\}$; moreover $\mu_a = \{v_a\}$ and $\mu_b = \{v_b\}$. See Fig. 4.19(a).

Note that, in any planar embedding of G, there exists a set S of at least $\frac{n}{6}$ nested 3-cycles, and all such cycles contain either v_a or v_b, say v_b, in their interior. Let C_i be any of such cycles. For each cluster μ containing a_i, b_i, and c_i, not all the edges of C_i can be drawn entirely inside the region $R(\mu)$ representing μ in any $\langle 0, \infty, 0\rangle$-drawing of $C(G,T)$, as otherwise $R(\mu)$ would enclose v_b. This implies that C_i intersects the border of $R(\mu)$ twice, hence creating an edge-region crossing. Since there exist $\Omega(n)$ cycles in S, each of which is contained in $\Omega(n)$ clusters, we have that any $\langle 0, \beta, 0\rangle$-drawing of $C(G,T)$ has $\beta = \Omega(n^2)$ edge-region crossings. □

Theorem 4.10 *There exists a c-connected flat clustered graph $C(G,T)$ such that $\beta \in \Omega(n)$ in every $\langle 0, \beta, 0\rangle$-drawing of $C(G,T)$.*

Proof: The underlying graph G is defined as in the proof of Theorem 4.9. Tree T is such that, for $i = 1, \ldots, n$, there exists a cluster μ_i containing vertices a_i, b_i, and c_i; moreover, $\mu_a = \{v_a\}$ and $\mu_b = \{v_b\}$. See Fig. 4.19(b).



(a) (b)

Figure 4.19: (a) Illustration for Theorem 4.9. (b) Illustration for Theorem 4.10.

In any planar embedding of G there exists a set S of at least $\frac{n}{6}$ nested 3-cycles, and all such cycles contain either v_a or v_b, say v_b, in their interior. Let C_i be any of such cycles. Not all the edges of C_i can be drawn entirely inside the region $R(\mu_i)$ representing μ_i in any $\langle 0, \infty, 0\rangle$-drawing of $C(G,T)$, as otherwise $R(\mu_i)$ would enclose v_b. This implies that C_i intersects the border of $R(\mu_i)$ twice. Since there exist $\Omega(n)$ cycles in S, we have that any $\langle 0, \beta, 0\rangle$-drawing of $C(G,T)$ has $\beta = \Omega(n)$ edge-region crossings. □

Finally, we prove lower bounds on the number of rr-crossings in $\langle 0, 0, \infty\rangle$-drawings of clustered graphs. We only consider non-c-connected clustered graphs, since a c-connected clustered graph either does not admit any $\langle 0, 0, \infty\rangle$-drawing or is c-planar. We distinguish two cases based on whether the considered clustered graphs are flat or not.

Theorem 4.11 *There exists a non-c-connected flat clustered graph $C(G,T)$, where G is outerplanar, such that $\gamma \in \Omega(n^2)$ in every $\langle 0, 0, \gamma\rangle$-drawing of $C(G,T)$.*

Proof: We first describe $C(G,T)$. Refer to Fig. 4.20(a). Consider a cycle $\mathcal{C}$ of n vertices $v_1, \ldots, v_n$, such that n is even. For $i = 1, \ldots, n$, add to $\mathcal{C}$ a vertex u_i and connect it to v_i and v_{i+1}, where $v_{n+1} = v_1$. Denote by G the resulting outerplanar graph. Tree T is such that vertices $v_1, \ldots, v_n$ belong to the same cluster μ^* and, for $i = 1, \ldots, n/2$, vertices u_i and $u_{n/2+i}$ belong to μ_i.

Since all vertices $u_1, \ldots, u_n$ have to lie outside region $R(\mu^*)$ in any $\langle 0, 0, \infty\rangle$-drawing of $C(G,T)$, the embedding of G is outerplanar. Hence, for any $i \neq j \in \{1, \ldots, n/2\}$, cluster μ_i intersects cluster μ_j, thus proving the theorem. □

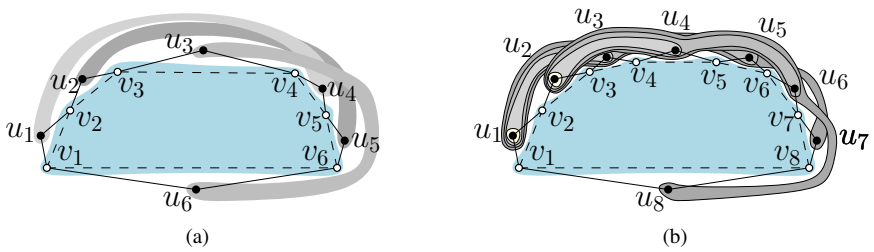


Figure 4.20: (a) Illustration for Theorem 4.11. (b) Illustration for Theorem 4.12.

Theorem 4.12 *There exists a non-c-connected non-flat clustered graph $C(G,T)$, where G is outerplanar, such that $\gamma \in \Omega(n^3)$ in every $\langle 0,0,\gamma\rangle$-drawing of $C(G,T)$.*

Proof: We first describe $C(G,T)$. Refer to Fig. 4.20(b). Graph G is an outerplanar graph constructed as in the proof of Theorem 4.11, such that n is a multiple of 4. Tree T is defined as follows. Set $\mu_1 = \{u_1\}$ and $\mu_2 = \{u_2\}$. Then, for each $i = 3, 4, \ldots, n$, set $\mu_i = \mu_{i-2} \cup \{u_i\}$. Finally, set $\mu^* = \{v_1, \ldots, v_n\}$.

Since all vertices $u_1, \ldots, u_n$ have to lie outside region $R(\mu^*)$, in any $\langle 0,0,\infty\rangle$-drawing of $C(G,T)$ the embedding of G is outerplanar.

We claim that, for each $i \in \{\frac{n}{2}, \frac{n}{2}+2, \ldots, n\}$ and $j \in \{\frac{n}{2}+1, \frac{n}{2}+3, \ldots, n-1\}$, the border of region $R(\mu_i)$ intersects $\Omega(n)$ times the border of region $R(\mu_j)$. Observe that the claim implies the theorem.

We prove the claim. Consider the border $B(\mu_i)$ of $R(\mu_i)$, for any $i \in \{\frac{n}{2}, \frac{n}{2}+2, \ldots, n\}$. First, for each $2 \leq k \leq \frac{n}{2}$ such that k is even, $B(\mu_i)$ properly crosses edge (v_k, u_k) in a point p_k and edge (v_{k+1}, u_k) in a point p'_k, given that μ_i contains u_k and does not contain v_k and v_{k+1}. Second, for each $1 \leq h \leq \frac{n}{2}$ such that h is odd, $B(\mu_i)$ does not cross edges (v_h, u_h), given that μ_i contains neither u_h, nor v_h, nor v_{k+1}. Third, the intersection point of $B(\mu_i)$ with G that comes after p_k and p'_k is p_{k+2}, as otherwise $B(\mu_i)$ would not be a simple curve or an er-crossing would occur. Analogous considerations hold for each $j \in \{\frac{n}{2}+1, \frac{n}{2}+3, \ldots, n-1\}$. Hence, the part of $B(\mu_i)$ between p'_k and p_{k+2} not containing p_k intersects the part of $B(\mu_j)$ between p'_{k+1} and p_{k+3}. This concludes the proof of the theorem. □

4.5 Relationships between α, β and γ

In this section we discuss the interplay between ee-, er-, and rr-crossings for the realizability of $\langle \alpha, \beta, \gamma\rangle$-drawings of clustered graphs.

As a first observation in this direction, we note that the result proved in Theorem 4.6 shows that there exist c-graphs for which allowing ee-, er-, and rr-crossing at the same time does not reduce the total number of crossings with respect to allowing only one type of crossings.

Next, we study the following question: suppose that a clustered graph $C(G,T)$ admits a $\langle 1,0,0\rangle$-drawing (resp. a $\langle 0,1,0\rangle$-drawing, resp. a $\langle 0,0,1\rangle$-drawing); does this imply that $C(G,T)$ admits a $\langle 0,\beta,0\rangle$-drawing and a $\langle 0,0,\gamma\rangle$-drawing (resp. an $\langle \alpha,0,0\rangle$-drawing and a $\langle 0,0,\gamma\rangle$-drawing, resp. an $\langle \alpha,0,0\rangle$-drawing and a $\langle 0,\beta,0\rangle$-drawing) with small number of crossings?

In the following, we prove that the answer to this question is often negative, as we can only prove (Theorem 4.13) that every graph admitting a drawing with one single er-crossing also admits a drawing with $O(n)$ ee-crossings, while in many other cases we can prove (Theorem 4.14) the existence of graphs that, even admitting a drawing with one single crossing of one type, require up to a quadratic number of crossings of a different type.

We first present Theorem 4.13. Observe that this theorem gives a stronger result than the one needed to answer the above question, as it proves that every $\langle \alpha,\beta,\gamma\rangle$-drawing of a clustered graph can be transformed into a $\langle \alpha+\beta\cdot O(n),0,\gamma\rangle$-drawing.

Theorem 4.13 *Any n-vertex clustered graph admitting a $\langle 0,\beta,0\rangle$-drawing also admits an $\langle \alpha,0,0\rangle$-drawing with $\alpha \in O(\beta n)$.*

Proof: Let Γ be a $\langle 0,\beta,0\rangle$-drawing of a clustered graph $C(G,T)$. We construct an $\langle \alpha,0,0\rangle$-drawing of $C(G,T)$ with $\alpha \in O(\beta n)$ by modifying Γ as follows. For each cluster $\mu \in T$, consider the set of edges that cross the boundary of $R(\mu)$ at least twice. Partition this set into two sets E_{in} and E_{out} as follows. Each edge whose endvertices both belong to μ is in E_{in}; each edge none of whose endvertices belongs to μ is in E_{out}; all the other edges are arbitrarily placed either in E_{in} or in E_{out}. Fig. 4.21(a) represents a cluster μ and the corresponding set E_{out}. We describe the construction for E_{out}. For each edge $e \in E_{out}$ consider the set of curves obtained as $e \cap R(\mu)$, except for the curves having the endvertices of e as endpoints. Consider the set $\mathcal{S}$ that is the union of the sets of curves obtained from all the edges of E_{out}. Starting from any point of the boundary of $R(\mu)$, follow such a boundary in clockwise direction and assign increasing integer labels to the endpoints of all the curves in $\mathcal{S}$. See Fig. 4.21(b). Consider a curve $\zeta \in \mathcal{S}$ such that there exists no other curve $\zeta' \in \mathcal{S}$ whose both endpoints have a label that is between the labels of the two endpoints of ζ. Then, consider the edge e such that ζ is a portion of e. Consider two points p_1 and p_2 of e arbitrarily close to the two endpoints of ζ, respectively, and not contained into $R(\mu)$. Redraw the portion of e between p_1 and p_2 as a curve outside $R(\mu)$ following

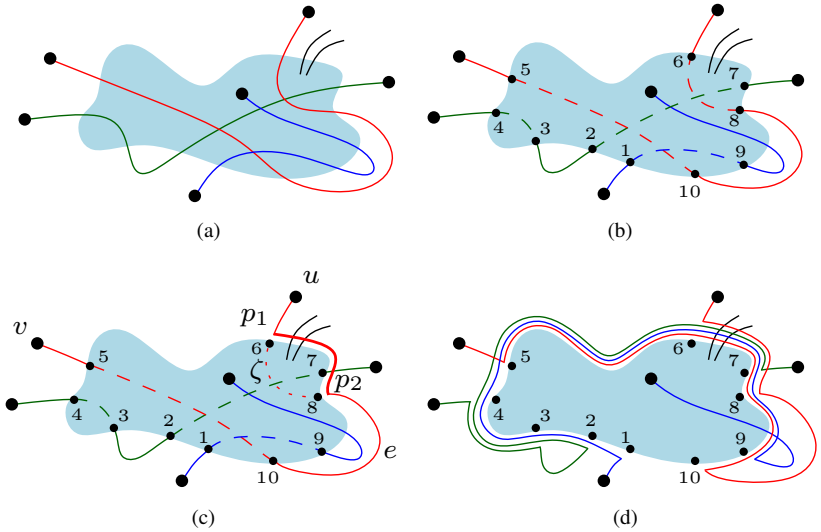


Figure 4.21: Illustration for Theorem 4.13. (a) A cluster μ with a set of edges crossing $R(\mu)$ at least twice and belonging to E_{out}. (b) The curves belonging to $\mathcal{S}$ are represented by dashed curve segments, while the other portions of the edges are represented by solid curve segments. The intersection points between curves in $\mathcal{S}$ and $R(\mu)$ are labeled with increasing integers. (c) The curve ζ between intersection points 6 and 8 that is a portion of edge $e = (u, v)$ is selected, since there exists no curve $\zeta' \in \mathcal{S}$ whose both endpoints have a label that is between 6 and 8. The old drawing of curve ζ is represented by a dotted curve segment, while the new drawing of ζ is represented by a fat solid curve. Note that the new drawing of ζ crosses all the edges that cross the boundary of $R(\mu)$ between 6 and 8. (d) The final drawing obtained by applying the described procedure to all the curves in $\mathcal{S}$.

clockwise the boundary $B(\zeta, \mu)$ of $R(\mu)$ between the smallest and the largest endpoint of ζ, and arbitrarily close to $B(\zeta, \mu)$ in such a way that it crosses only the edges that cross $B(\zeta, \mu)$ and the edges that used to cross the portion of e between p_1 and p_2 before redrawing it. See Fig. 4.21(c), where the curve ζ between 6 and 8 is redrawn. Remove ζ from $\mathcal{S}$ and repeat this procedure until $\mathcal{S}$ is empty. Fig. 4.21(d) shows the final drawing obtained by applying the described procedure to the drawing in Fig. 4.21(a).

The construction for E_{in} is analogous, with the portion of e being redrawn inside

$R(\mu)$. Observe that, every time the portion of an edge e between p_1 and p_2, corresponding to a curve $\zeta \in \mathcal{S}$, is redrawn, an er-crossing is removed from the drawing and at most $O(n)$ ee-crossings between e and the edges crossing $B(\zeta, \mu)$ are added to the drawing. This concludes the proof of the theorem. □

In Theorem 4.14 we prove that there exist graphs that, even admitting a drawing with one single crossing of one type, require up to a quadratic number of crossings of a different type.

Theorem 4.14 *There exist clustered graphs C_1, C_2, and C_3 such that:*

(i) C_1 admits a $\langle 1,0,0\rangle$-drawing, $\beta \in \Omega(n^2)$ in every $\langle 0,\beta,0\rangle$-drawing of C_1, and $\gamma \in \Omega(n^2)$ in every $\langle 0,0,\gamma\rangle$-drawing of C_1;

(ii) C_2 admits a $\langle 0,1,0\rangle$-drawing, $\alpha \in \Omega(n)$ in every $\langle \alpha,0,0\rangle$-drawing of C_2, and $\gamma \in \Omega(n^2)$ in every $\langle 0,0,\gamma\rangle$-drawing of C_2;

(iii) C_3 admits $\langle 0,0,1\rangle$-drawing, $\alpha \in \Omega(n^2)$ in every $\langle \alpha,0,0\rangle$-drawing, and $\beta \in \Omega(n)$ in every $\langle 0,\beta,0\rangle$-drawing of C_3.

Proof: We start by describing a clustered graph $C^*(G^*, T^*)$, that will be used as a template for the graphs in the proof. Graph G^* is obtained as follows.Refer to Fig. 4.22(a). Initialize $G^* = K_5(n)$ on vertices $\{a, b, c, d, e\}$. First, for each $u, v \in \{a, b, c, d, e\}$, with $u \neq v$, replace the set of n multiple edges $S(u, v)$ with a set $S'(u, v)$ of n length-2 paths between u and v. Then, remove from G^* sets $S'(a, d)$, $S'(c, e)$, $S'(a, e)$, and $S'(c, d)$. Finally, for $i = 1, \ldots, n$, add to G^* vertices $[ae]_i, [ea]_i, [cd]_i, [dc]_i$, and edges $(a, [ae]_i)$, $(e, [ea]_i)$, $(c, [cd]_i)$, $(d, [dc]_i)$. For $i = 1, \ldots, m$, T^* contains clusters $\mu(a, e)_i = \{[ae]_i, [ea]_i\}$ and $\mu(c, d)_i = \{[cd]_i, [dc]_i\}$. Denote by $M(a, e) = \{(a, [ae]_i), (e, [ea]_i), \mu(a, e)_i | i = 1, \ldots, n\}$ and $M(c, d) = \{(c, [cd]_i), (d, [dc]_i), \mu(c, d)_i | i = 1, \ldots, n\}$.

Clustered graph $C_1(G_1, T_1)$ is obtained by adding edges (a, d) and (c, e) to G^* and by setting $T_1 = T^*$. A $\langle 1, 0, 0\rangle$-drawing of C_1 is depicted in Fig. 4.22(b), where edges (a, d) and (c, e) cross.

Clustered graph $C_2(G_2, T_2)$ is obtained by adding edge (c, e) to G^* and by adding a cluster $\mu(a, d) = \{a, d\}$ to T^*. A $\langle 0, 1, 0\rangle$-drawing of C_2 is depicted in Fig. 4.22(c), where edge (c, e) and region $R(\mu(a, d))$ cross.

Clustered graph $C_3(G_3, T_3)$ is obtained by setting $G_3 = G^*$ and by adding clusters $\mu(a, d) = \{a, d\}$ and $\mu(c, e) = \{c, e\}$ to T^*. A $\langle 0, 0, 1\rangle$-drawing of C_3 is depicted in Fig. 4.22(d), where regions $R(\mu(a, d))$ and $R(\mu(c, e))$ cross.

The lower bounds claimed in the theorem can be obtained with arguments analogous to those used in the proof of Theorem 4.6.

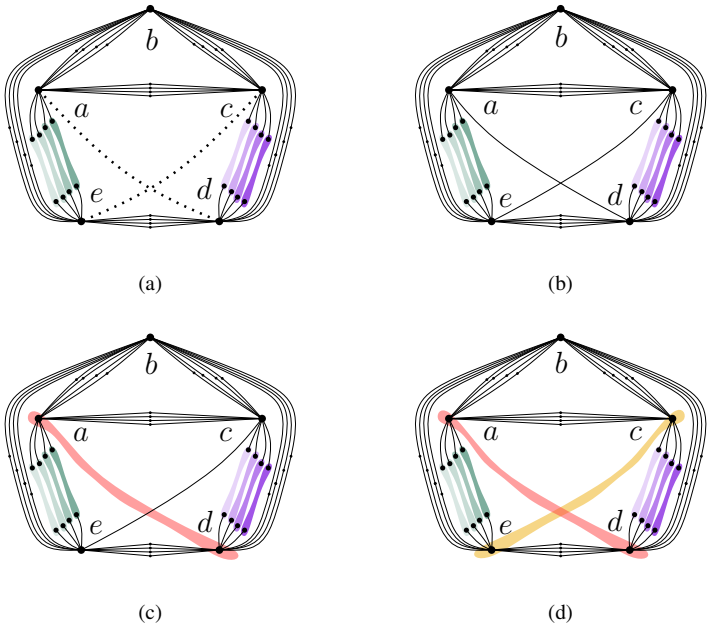


Figure 4.22: Illustration for the proof of Theorem 4.14. (a) C-graph C^*: Dotted lines are placeholders for gadgets. (b) A $\langle 1, 0, 0\rangle$-drawing of C_1, (c) a $\langle 0, 1, 0\rangle$-drawing of C_2, and (d) a $\langle 0, 0, 1\rangle$-drawing of C_3.

For example, consider any $\langle 0, \beta, 0\rangle$-drawing Γ_β of C_1 which minimizes β. For $i = 1, \ldots, n$, draw an edge $([ae]_i, [ea_i])$ inside region $R(\mu(a, e)_i)$ and an edge $([cd]_i, [dc_i])$ inside region $R(\mu(c, d)_i)$, and remove such regions. Then, remove edges (a, d) and (c, e) and draw two sets $S(a, d)$ and $S(c, e)$ of n multiple edges arbitrarily close to the drawings of (a, d) and (c, e). The obtained drawing Γ'_β is a drawing of a subdivision of $K_5(n)$, and hence contain $\Omega(n^2)$ crossings. Since Γ_β is a $\langle 0, \infty, 0\rangle$-drawing, each crossing in Γ'_β involves exactly one edge in $\{([ae]_i, [ea_i]), ([cd]_i, [dc_i])\}$. Also, since Γ_β minimizes β, edges in $\{(a, d), (c, e)\}$ are not involved in any crossing in Γ'_β, since both such edges are adjacent to an edge belonging to $M(a, e)$ and to an edge belonging to $M(c, d)$ (recall that adjacent edges do not cross in any drawing of a graph whose number of crossings is minimum). Thus, each crossing in Γ'_β corresponds to an er-crossing in Γ_β, which implies that $\beta \in \Omega(n^2)$.

This concludes the proof of the theorem. □

4.6 Complexity

In this section we study the problem of minimizing the number of crossings in $\langle\alpha, \beta, \gamma\rangle$-drawings.

We define the problem (α, β, γ)-CLUSTERCROSSINGNUMBER((α, β, γ)-CCN) as follows. Given a clustered graph $C(G, T)$ and an integer $k > 0$, problem (α, β, γ)-CCN asks whether $C(G, T)$ admits a $\langle\alpha, \beta, \gamma\rangle$-drawing with $\alpha + \beta + \gamma \leq k$.

First, we prove that problem (α, β, γ)-CCN belongs to class NP.

Lemma 4.7 *Problem (α, β, γ)-CCN is in NP.*

Proof: Similarly to the proof that the CROSSINGNUMBER problem is in NP [GJ83], we need to "guess" a drawing of $C(G, T)$ with α ee-crossings, with β er-crossings, and with γ rr-crossings, for each choices of α, β, and γ satisfying $\alpha + \beta + \gamma \leq k$. This is done as follows. Let m be the number of edge-cluster pairs $\langle e, \mu\rangle$ such that one end-vertex of e is in μ and the other one is not. Let $0 \leq p \leq \gamma$ be a guess on the number of pairs of clusters that intersect each other. Let $\mathcal{E}$ be a guess on the rotation schemes of the vertices of G. Arbitrarily orient each edge in G; also, arbitrarily fix a "starting point" on the boundary of each cluster in T and orient such a boundary in any way.

For each edge e, guess a sequence of crossings $x_1, x_2, \ldots, x_{k(e)}$ occurring along e while traversing it according to its orientation. Each of such crossings x_i is associated with: (1) the edge e' that crosses e in x_i or the cluster μ' such that the boundary of $R(\mu')$ crosses e in x_i; and (2) a boolean value $b(x_i)$ stating whether e' (resp. the boundary of $R(\mu')$) crosses e from left to right according to the orientations of e and e' (resp. of e and the boundary of $R(\mu')$).

Analogously, for each cluster μ, guess a sequence of crossings $x_1, x_2, \ldots, x_{k(\mu)}$ occurring along the boundary of $R(\mu)$ while traversing it from its starting point according to its orientation. Again, each of such crossings x_i is associated with: (1) the edge e' that crosses the boundary of $R(\mu)$ in x_i or the cluster μ' such that the boundary of $R(\mu')$ crosses the boundary of $R(\mu)$ in x_i; and (2) a boolean value $b(x_i)$ stating whether e' (resp. the boundary of $R(\mu')$) crosses the boundary of $R(\mu)$ from left to right according to the orientations of the boundary of $R(\mu)$ and e' (resp. of the boundary of $R(\mu)$ and the boundary of $R(\mu')$). Observe that the guessed crossings respect constraints $\mathcal{C}_1$, $\mathcal{C}_2$, and $\mathcal{C}_3$.

Crossings are guessed in such a way that there is a total number of α crossings between edge-edge pairs, a total number of $2\beta + 2m$ crossings between edge-cluster

pairs, and a total number of $2\gamma + 2p$ crossings between cluster-cluster pairs, so that p pairs of clusters have a crossing.

We construct a graph G^* with a fixed rotation scheme around each vertex as follows. Start with G^* having the same vertex set of G and containing no edge. For each edge e in G, add to G^* a path starting at one end-vertex of e, ending at the other end-vertex of e, and containing a vertex for each crossing associated with e. For each cluster μ in T, add to G^* a cycle containing a vertex for each crossing associated with μ. This is done in such a way that one single vertex is introduced in G^* for each guessed crossing. The rotation scheme of each vertex in G^* that is also a vertex in G is the one in $\mathcal{E}$. The rotation scheme of each vertex in G^* corresponding to a crossing x_i is determined according to $b(x_i)$.

Check in linear time whether the constructed graph G^* with a fixed rotation scheme around each vertex is planar. For each cluster μ, check in linear time whether the cycle representing the boundary of $R(\mu)$ contains in its interior all and only the vertices of G and the clusters in T (that is, all the vertices of the cycles representing such clusters) it has to contain. Observe that, if the checks succeed and a planar drawing of G^* with the a fixed rotation scheme around each vertex can be constructed, the corresponding drawing of $C(G,T)$ is an $\langle\alpha,\beta,\gamma\rangle$-drawing. □

Second, we prove that (α,β,γ)-CCN is NP-complete, even if the underlying graph is planar, namely a forest of star graphs, by means of a reduction from the CROSSINGNUMBER problem.

Theorem 4.15 *Problem (α,β,γ)-CCN is NP-complete, even in the case in which the underlying graph is a forest of star graphs.*

Proof: The membership in NP is proved in Lemma 4.7.

The NP-hardness is proved by means of a polynomial-time reduction from the CROSSINGNUMBER problem, which has been proved to be NP-complete by Garey and Johnson [GJ83]. Given a graph G^* and an integer $k^* > 0$, the CROSSINGNUMBER problem consists of deciding whether G^* admits a drawing with at most k^* crossings.

We describe how to construct an instance $\langle C(G,T), k\rangle$ of (α,β,γ)-CCN starting from an instance $\langle G^*, k^*\rangle$ of CROSSINGNUMBER.

Initialize $G = G^*$ and $T = \rho$. For each edge (u_i,u_j) of G^*, subdivide (u_i,u_j) with two subdivision vertices u'_i and u'_j, add a cluster $\mu_{i,j}$ to T containing u'_i and u'_j as a child of ρ, and remove edge (u'_i,u'_j) from G and from G^*. See Fig.4.23. Further, set $k = k^*$. Note that, graph G is a forest of star graphs. Also, instance $\langle$C(G,T) , $k\rangle$ can be constructed in polynomial time.

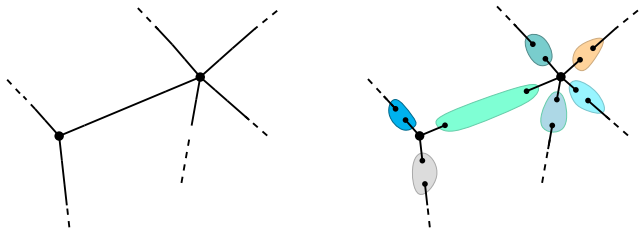


Figure 4.23: Illustration for the proof of Theorem 4.15: A part of graph G^* (a) and the corresponding part of $C(G,T)$ (b).

We show that instance $\langle C(G,T),k\rangle$ has a solution if and only if instance $\langle G^*,k^*\rangle$ has a solution. Both directions of the proof use techniques similar to those used in Section 4.4.

Suppose that $\langle G^*,k^*\rangle$ admits a solution, that is, G^* has a drawing Γ^* with at most k^* crossings. An $\langle\alpha,\beta,\gamma\rangle$-drawing Γ of $C(G,T)$ with $\alpha+\beta+\gamma\leq k$ can be constructed by subdividing twice each edge (u_i,u_j) of G^* and by replacing the central edge (u'_i,u'_j) of each length-3 path representing an edge (u_i,u_j) of G^* with a cluster whose drawing is arbitrarily close to the drawing of (u'_i,u'_j). By construction, each crossing between two edges in Γ^* corresponds to either an ee-crossing, or to an er-crossing, or to a rr-crossing in Γ. Hence, drawing Γ contains the same number of crossings as Γ^*, that is, at most $k^*=k$.

Suppose that $\langle C(G,T),k\rangle$ admits a solution, that is, $C(G,T)$ has an $\langle\alpha,\beta,\gamma\rangle$-drawing Γ with $\alpha+\beta+\gamma\leq k$. A drawing Γ^* of G^* with at most k^* crossings can be constructed by replacing each cluster $\mu_{i,j}=\{u'_i,u'_j\}$ with an edge between u'_i and u'_j inside $R(\mu_{i,j})$ and by replacing each length-3 path $P(i,j)=(u_i,u'_i,u'_j,u_j)$ with an edge (u_i,u_j) whose drawing is the same as the drawing of $P(i,j)$ in Γ. By construction, each crossing (that is either an ee-crossing, or an er-crossing, or an rr-crossing) in Γ corresponds to a crossing between two edges in Γ^*. Hence, drawing Γ^* contains the same number of crossings as Γ, that is, at most $k=k^*$.

This concludes the proof of the theorem. □

As for the problems considered in the previous sections, it is interesting to study the (α,β,γ)-CCN problem when only one out of α, β, and γ is allowed to be different from 0. We call α-CCN, β-CCN, and γ-CCN the corresponding decision problems.

We observe that the proof of Theorem 4.15 can be easily modified to show that all of α-CCN, β-CCN, and γ-CCN are NP-complete, even in the case in which the

underlying graph is a forest of star graphs.

In the following we prove that stronger results can be found for α-CCN and β-CCN, by giving NP-hardness proofs for more restricted clustered graph classes.

Theorem 4.16 *Problem α-CCN is NP-complete even in the case in which the underlying graph is a matching.*

Proof: The membership in NP follows from Lemma 4.7.

The NP-hardness is proved by means of a polynomial-time reduction from the known CROSSINGNUMBER problem [GJ83].

We describe how to construct an instance $\langle$C(G,T) , $k\rangle$ of α-CCN starting from an instance $\langle G^*, k^*\rangle$ of CROSSINGNUMBER. See Figs. 4.24(a)-(b).

Initialize $G = G^*$ and $T = \rho$. Subdivide each edge of G with two subdivision vertices. For each vertex v_i of G, add a cluster μ_i to T as a child of ρ containing all the neighbors of v_i, and remove from G vertex v_i and its incident edges.

Further, set $k = k^*$. Note that graph G is a matching. Also, instance $\langle C(G,T), k\rangle$ can be constructed in polynomial time.

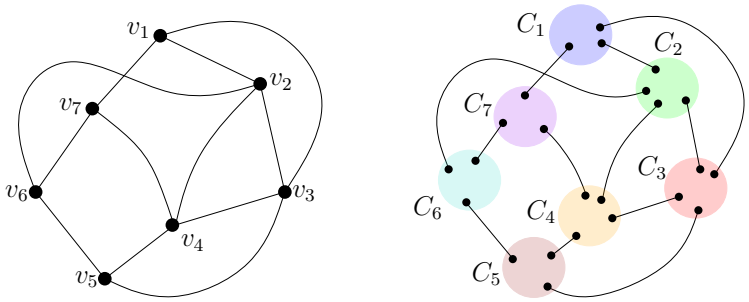


Figure 4.24: (a) Graph G^* in the proof of Theorem 4.16. (b) The clustered graph $C(G,T)$ corresponding to G^*.

We show that instance $\langle C(G,T), k\rangle$ has a solution if and only if instance $\langle G^*, k^*\rangle$ has a solution.

Suppose that $\langle G^*, k^*\rangle$ admits a solution, that is, G^* has a drawing Γ^* with at most k^* crossings. An $\langle\alpha, 0, 0\rangle$-drawing Γ of $C(G,T)$ with $\alpha \leq k$ can be constructed as follows. Initialize $\Gamma = \Gamma^*$. For each vertex v_i of G^*, draw a small disk d_i around it, and, for each edge e incident to v_i, place a vertex v_i' on the intersection between

e and the boundary of d_i. Then, replace each edge $e = (v_i, v_j)$ in Γ with edge (v'_i, v'_j). Finally, remove each vertex v_i of G^* from Γ and represent each cluster μ_i in Γ as a region slightly surrounding disk d_i. Since each edge in Γ is represented as a Jordan curve that is a subset of the Jordan curve representing an edge in Γ^*, drawing Γ contains at most the same number of crossings as Γ^*.

Suppose that $\langle C(G,T), k\rangle$ admits a solution, that is, $C(G,T)$ has an $\langle \alpha, 0, 0\rangle$-drawing Γ with $\alpha \leq k$. A drawing Γ^* of G^* with at most k^* crossings can be constructed as follows. Place vertex v_i on any interior point of region $R(\mu_i)$. For each intersection point p between the boundary of $R(\mu_i)$ and an edge incident to μ_i, draw a curve connecting v_i to p so that such curves do not cross each other. Remove each vertex v'_i and, for each edge e incident to v'_i, the part of e which lies inside $R(\mu_i)$. Also, remove all the regions representing clusters of T. The crossings in the resulting drawing Γ^* of G^* are a subset of the ee-crossings in Γ. Namely, the curves that exist in Γ^* and do not exist in Γ do not cross any edge of G^*, given that Γ has no er-crossing. This concludes the proof of the theorem. □

Theorem 4.17 *Problem β-CCN is NP-complete even for c-connected flat clustered graphs in which the underlying graph is a triconnected planar multigraph.*

Proof: The membership in NP follows from Lemma 4.7.

The NP-hardness is proved by means of a polynomial-time reduction from the NP-complete problem PLANAR STEINER TREE (STPG) [GJ77] (see also Section 8.4 for further results on the STPG problem contained in this thesis), which is defined as follows: Given a planar graph $G(V,E)$ whose edges have weights $w : E \to \mathbb{N}$, given a set $S \subset V$ of *terminals*, and given an integer k, the STPG problem asks whether there exists a tree $T^*(V^*, E^*)$ such that (1) $V^* \subseteq V$, (2) $E^* \subseteq E$, (3) $S \subseteq V^*$, and (4) $\sum_{e \in E^*} w(e) \leq k$. The edge weights in w are bounded by a polynomial function $p(n)$ (see [GJ77]). We are going to use the variant of STPG in which (A) G is a subdivision of a triconnected planar graph, where each subdivision vertex is not a terminal, and (B) all the edge weights are equal to 1.

In the following we sketch a reduction from STPG to STPG with the described properties. Let G be any edge-weighted planar graph. Augment G to any triconnected planar graph $G'(V', E')$ by adding dummy edges and by assigning weight $w(e) = n \cdot p(n)$ to each dummy edge e. Then, replace each edge e of G' with a path $P(e)$ with $w(e)$ edges, each with weight 1, hence obtaining a planar graph $G''(V'', E'')$. Let the terminals of G'' be the same terminals of G. Note that, by construction, G'' satisfies Properties (A) and (B). Also, since $|V''| \in O(n^2 \cdot p(n))$, the described reduction is polynomial.

We prove that $\langle G, S, k\rangle$ is a positive instance of STPG if and only if $\langle G'', S, k\rangle$ is a positive instance of the considered variant of STPG.

Suppose that $\langle G, S, k\rangle$ is a positive instance of STPG. Starting from the solution T^* of $\langle G, S, k\rangle$, we construct a solution $T^\diamond$ of $\langle G'', S, k\rangle$ by replacing each edge e with path $P(e)$. By construction, $\sum_{e\in T^\diamond} w(e) = \sum_{e\in T^*} w(e) \leq k$.

Suppose that $\langle G'', S, k\rangle$ is a positive instance of the variant of STPG. Let $T^\diamond(V^\diamond, E^\diamond)$ be the solution of $\langle G'', S, k\rangle$. Assume that $T^\diamond$ is the *optimal* solution to $\langle G'', S, k\rangle$, i.e., there exists no tree $T^\sharp(V^\sharp, E^\sharp)$ such that $T^\sharp(V^\sharp, E^\sharp)$ is a solution to $\langle G'', S, k\rangle$ and $\sum_{e\in E^\sharp} w(e) < \sum_{e\in E^\diamond} w(e)$. Observe that, if an edge of a path $P(e)$ belongs to $E^\diamond$, then all the edges of $P(e)$ belong to $E^\diamond$. Moreover, no edge of a path $P(e)$ such that e is a dummy edge belongs to $E^\diamond$, since the total weight of the edges of each path $P(e)$ such that e is a dummy edge is $n \cdot p(n)$. Starting from $T^\diamond$, we construct a solution T^* of $\langle G, S, k\rangle$ by replacing all the edges of each path $P(e)$ with an edge e. By construction, $\sum_{e\in T^*} w(e) = \sum_{e\in T^\diamond} w(e) \leq k$.

Next we show a polynomial-time reduction from the variant of STPG in which all the instances satisfy Properties (A) and (B) to β-CCN. Refer to Fig. 4.25. Let $\langle G, S, k\rangle$ be an instance of the variant of STPG. Since G is a subdivision of a triconnected planar graph, it admits a unique planar embedding, up to a flip and to the choice of the outer face. Construct a planar embedding Γ_G of G. Construct the dual graph H of Γ_G. Note that, since G is a subdivision of a triconnected planar graph, its dual H is a planar triconnected multigraph. For each terminal $s \in S$, consider the set $E_G(s)$ of the edges incident to s in G and consider the face f_s of H composed of the edges that are dual to the edges in $E_G(s)$; add s to the vertex set of H, embed it inside f_s, and connect it to the vertices incident to f_s. Define the inclusion tree T as follows. For each vertex $s_i \in S$, with $1 \leq i \leq |S|$, T has a cluster $\mu_i = \{s_i\}$; all the other vertices in the vertex set of H belong to the same cluster ν. Then, the instance of β-CCNis $\langle C(H, T), k\rangle$.

We show that $\langle C(H, T), k\rangle$ admits a solution if and only if $\langle G, S, k\rangle$ does.

Suppose that $\langle G, S, k\rangle$ admits a solution T^*. Consider a terminal vertex $s^* \in S$ and construct a planar embedding of H such that s^* is incident to the outer face. Construct a drawing of cluster ν as a simple region $R(\nu)$ that entirely encloses H, except for a small region surrounding T^* (observe that such a simple region $R(\nu)$ exists since s^* is incident to the outer face and $s^* \in T^*$). Draw each cluster μ_i as a region $R(\mu_i)$ surrounding s_i sufficiently small so that it does not intersect $R(\nu)$. Observe that the resulting drawing of $C(H, T)$ is a $\langle 0, \infty, 0\rangle$-drawing. Moreover, $R(\nu)$ intersects all and only the edges dual to edges in T^*, hence there are at most k edge-region crossings, that is, $C(H, T)$ is a $\langle 0, \beta, 0\rangle$-drawing with $\beta \leq k$.

Suppose that $C(H, T)$ admits a $\langle 0, \beta, 0\rangle$-drawing Γ with $\beta \leq k$ edge-region crossings and assume that Γ is optimal (that is, there is no $\langle 0, \infty, 0\rangle$-drawing with fewer

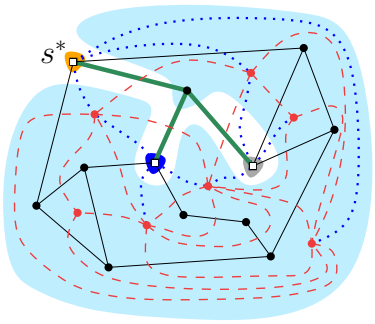


Figure 4.25: Illustration for the proof of Theorem 4.17. Solid (black) lines are edges of G; dashed (red) and dotted (blue) lines are edges of H; green edges are the edges of T^*; black circles and white squares are non-terminal vertices and terminals in G, respectively; finally, red circles and white squares are vertices in H.

er-crossings). Consider the graph T^* composed of the edges that are dual to the edges of H participating in some edge-region crossing. We claim that T^* has at least one edge incident to each terminal in S and that T^* is connected. The claim implies that T^* is a solution to the instance $\langle G, S, k\rangle$ of STPG, since T^* has at most k edges and since Γ is optimal. Consider any terminal $s \in S$. If none of the edges incident to s in G belongs to T^*, it follows that none of the edges of H incident to face f_s has a crossing with the region $R(\nu)$ representing ν in Γ. Observe that, since Γ is optimal, there exists a terminal s^* incident to the outer face of Γ. If $s \neq s^*$, then since all the vertices incident to f_s have to lie inside $R(\nu)$, either $R(\nu)$ is not a simple region or it contains s, in both cases contradicting the assumption that Γ is a $\langle 0, \infty, 0\rangle$-drawing. Also, if $s = s^*$, then either $R(\nu)$ is not a simple region or it contains all the vertices of H, and hence also vertices not in ν, in both cases contradicting the assumption that Γ is a $\langle 0, \infty, 0\rangle$-drawing. Suppose that T^* contains (at least) two connected components T_1^* and T_2^*. At least one of them, say T_2^*, does not contain any edge that is dual to an edge incident to the outer face of Γ. Therefore, $R(\nu)$ is not a simple region, a contradiction. This concludes the proof of the theorem. □

4.7 Open Problems

Given a clustered graph whose underlying graph is planar we defined and studied its $\langle \alpha, \beta, \gamma\rangle$-drawings, where the number of ee-, er-, and rr-crossings is equal to α, β, and γ, respectively.

This chapter opens several problems. First, some of them are identified by non-tight bounds in the tables of the Introduction. Second, in order to study how allowing different types of crossings impacts the features of the drawings, we concentrated most of the attention on $\langle \alpha, \beta, \gamma \rangle$-drawings where two out of α, β, and γ are equal to zero. It would be interesting to study classes of clustered graphs that have drawings where the values of α, β, and γ are balanced in some way. Third, we have seen that not all clustered graphs whose underlying graph is planar admit $\langle 0, 0, \infty \rangle$-drawings. It would be interesting to characterize the class of clustered graphs that admit one and to extend our testing algorithm to simply-connected clustered graphs.

Chapter 5

Planar Embeddings with Small and Uniform Faces

Motivated by finding planar embeddings that lead to drawings with favorable aesthetics, in this chapter[1] we study the problems MINMAXFACE and UNIFORMFACES of embedding a given biconnected multi-graph such that the largest face is as small as possible and such that all faces have the same size, respectively. Our study is further justified by the fact that there exist many computationally challeging problems that have been proved hard even for planar graphs for which efficient algorithms are known only if the input graphs have low degree. In particular, the C-PLANARITY problem was proved polynomial-time solvable for plane clustered graphs with small faces [JKK$^+$09, DF09], or equivalently, whose dual graph has small degree.

We prove a complexity dichotomy for MINMAXFACE and show that deciding whether the maximum is at most k is polynomial-time solvable for $k \leq 4$ and NP-complete for $k \geq 5$. Further, we give a 6-approximation for minimizing the maximum face in a planar embedding. For UNIFORMFACES, we show that the problem is NP-complete for odd $k \geq 7$ and even $k \geq 10$. Moreover, we characterize the biconnected planar multi-graphs admitting 3- and 4-uniform embeddings (in a k-uniform embedding all faces have size k) and give an efficient algorithm for testing the existence of a 6-uniform embedding.

[1]The contents of this chapter are a joint work with Vít Jelínek, Jan Kratochvíl, and Ignaz Rutter, appeared in [LJKR14] and carried out during a visit period in the Department of Applied Mathematics at Charles University of Prague. Thanks to Bartosz Walczak for fruitful discussions.

5.1 Introduction

While there are infinitely many ways to embed a connected planar graph into the plane without edge crossings, these embeddings can be grouped into a finite number of equivalence classes, so-called *combinatorial embeddings*, where two embeddings are *equivalent* if the clockwise order around each vertex is the same. Many algorithms for drawing planar graphs require that the input graph is provided together with a combinatorial embedding, which the algorithm preserves. Since the aesthetic properties of the drawing often depend critically on the chosen embedding, e.g. the number of bends in orthogonal drawings, it is natural to ask for a planar embedding that will lead to the best results.

In many cases the problem of optimizing some cost function over all combinatorial embeddings is NP-complete. For example, it is known that it is NP-complete to test the existence of an embedding that admits an orthogonal drawing without bends or an upward planar embedding [GT01a]. On the other hand, there are efficient algorithms for minimizing various measures such as the radius of the dual [ADP11, BM88] and attempts to minimize the number of bends in orthogonal drawings subject to some restrictions [BKRW14, BRW13, DLV98].

Usually, choosing a planar embedding is considered as deciding the circular ordering of edges around vertices. It can, however, also be equivalently viewed as choosing the set of facial cycles, i.e., which cycles become face boundaries. In this sense it is natural to seek an embedding whose facial cycles have favorable properties. For example, Gutwenger and Mutzel [GM04] give algorithms for computing an embedding that maximizes the size of the outer face. The most general form of this problem is as follows. The input consists of a graph and a cost function on the cycles of the graph, and we seek a planar embedding where the sum of the costs of the facial cycles is minimum. This general version of the problem has been introduced and studied by Mutzel and Weiskircher [MW99]. Woeginger [Woe02] shows that it is NP-complete even when assigning cost 0 to all cycles of size up to k and cost 1 for longer cycles. Mutzel and Weiskircher [MW99] propose an ILP formulation for this problem based on SPQR-trees.

In this chapter, we focus on two specific problems of this type, aimed at reducing the visual complexity and eliminating certain artifacts related to face sizes from drawings. Namely, large faces in the interior of a drawing may be perceived as holes and consequently interpreted as an artifact of the graph. Similarly, if the graph has faces of vastly different sizes, this may leave the impression that the drawn graph is highly irregular. However, rather than being a structural property of the graph, it is quite possible that the artifacts in the drawing rather stem from a poor embedding choice and can be avoided by choosing a more suitable planar embedding.

We thus propose two problems. First, to avoid large faces in the drawing, we seek to minimize the size of the largest face; we call this problem MINMAXFACE. Second, we study the problem of recognizing those graphs that admit perfectly uniform face sizes; we call this problem UNIFORMFACES. Both problems can be solved by the ILP approach of Mutzel and Weiskircher [MW99] but were not known to be NP-hard.

Our Contributions. First, in Section 5.3, we study the computational complexity of MINMAXFACE and its decision version k-MINMAXFACE, which asks whether the input graph can be embedded such that the maximum face size is at most k. We prove a complexity dichotomy for this problem and show that k-MINMAXFACE is polynomial-time solvable for $k \leq 4$ and NP-complete for $k \geq 5$. Our hardness result for $k \geq 5$ strengthens Woeginger's result [Woe02], which states that it is NP-complete to minimize the number of faces of size greater than k for $k \geq 4$, whereas our reduction shows that it is in fact NP-complete to decide whether such faces can be completely avoided. Furthermore, we give an efficient 6-approximation for MINMAXFACE. Our result for $k \leq 4$ suggests a possible strategy to try to answer the C-PLANARITY problem in some cases. In fact, given an instance $C(G,T)$ of the C-PLANARITY problem one could first test whether the underlying graph G allows for an embedding $\mathcal{E}$ whose every faces have size at most 4 and then run one of the algorithms by Jelínková *et al.* [JKK$^+$09] or by Di Battista and Frati [DF09] to test the C-PLANARITY of $C(G,T)$ when the planar embedding of G is $\mathcal{E}$.

Second, in Section 5.4, we study the problem of recognizing graphs that admit perfectly uniform face sizes (UNIFORMFACES), which is a special case of k-MINMAXFACE. An embedding is *k-uniform* if all faces have size k. We characterize the biconnected multi-graphs admitting a k-uniform embedding for $k = 3, 4$ and give an efficient recognition algorithm for $k = 6$. Finally, we show that for odd $k \geq 7$ and even $k \geq 10$, it is NP-complete to decide whether a planar graph admits a k-uniform embedding.

5.2 Preliminaries

Unless specified otherwise, throughout the rest of the chapter we will consider graphs without loops, but with possible multiple edges.

To handle the decomposition of a biconnected multi-graph into its triconnected components we use SPQR-trees. Refer to Chapter 2 and to [DT90, DT96b, DT96a]. Recall that, planar embeddings of G correspond bijectively to planar embeddings of all skeletons of its SPQR-tree $\mathcal{T}$; the choices are the orderings of the parallel edges in P-nodes and the embeddings of the R-node skeletons, which are unique up to a flip.

When considering rooted SPQR-trees, we assume that the embedding of G is such that the root edge is incident to the outer face, which is equivalent to the parent edge being incident to the outer face in each skeleton. We refer the reader to Chapter 2 for the related concepts of *reference edge*, *pertinent graph*, *expansion graph*, *skeleton*, and *virtual edge*. We remark that in a planar embedding of G, the poles of any node μ of $\mathcal{T}$ are incident to the outer face of the pertinent graph $pert(\mu)$ of μ. Hence, in the following we only consider embeddings of the pertinent graphs with their poles lying on the same face.

5.3 Minimizing the Maximum Face

In this section we present our results on MINMAXFACE. We first strengthen the result of Woeginger [Woe02] and show that k-MINMAXFACE is NP-complete for $k \geq 5$ and then present efficient algorithms for $k = 3, 4$. In particular, the hardness result also implies that the problem MINMAXFACE is NP-hard. Finally, we give an efficient 6-approximation for MINMAXFACE on biconnected graphs. Recall that we allow graphs to have multiple edges.

Theorem 5.1 *k-MINMAXFACE is NP-complete for any $k \geq 5$.*

Proof: Clearly, the problem is in NP, since we can simply guess a planar embedding and verify in polynomial time that all faces have size at most k.

We show hardness for $k = 5$ and in the end briefly sketch how to adapt the proof for $k > 5$. We give a reduction from PLANAR 3-SAT with the additional assumption that each variable occurs three times and each clause has size two or three. Further, we can assume that if a variable occurs three times, then it appears twice as a positive literal and once as a negative literal. This variant is NP-complete [FKMP95, Lemma 2.1].

We construct gadgets where some of the edges are in fact two parallel paths, one consisting of a single edge and one of length 2 or 3. The ordering of these paths then decides which of the faces incident to the gadget edge is incident to a path of length 1 and which is incident to a path of length 2 or 3; see Fig. 5.1a. Due to this use, we also call these gadgets $(1, 2)$- and $(1, 3)$-edges, respectively.

Now consider a variable x whose positive literals occur d^+ times. Note that the negative literal hence occurs $3 - d^+$ times. We represent x by a *variable gadget* consisting of two cycles C^+ and C^- of lengths $5 - d^+$ and $5 - (3 - d^+) = 2 + d^+$, respectively, sharing one edge. The shared edge is actually a $(1, 3)$-edge, called *variable edge*, and in C^+ (in C^-), we replace d^+ of its edges ($3 - d^+$ of its edges) by $(1, 2)$-edges, called positive (negative) *literal edges*, respectively; see Fig. 5.1b. We

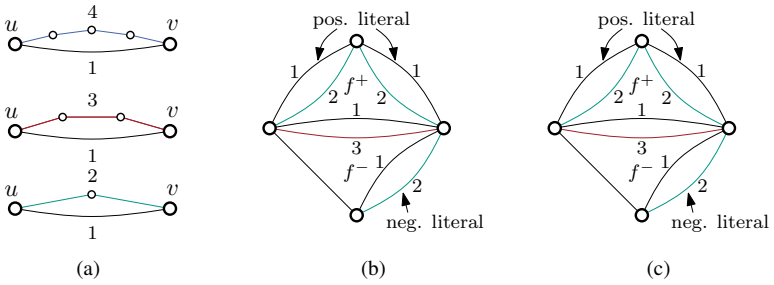


Figure 5.1: Illustration of the gadgets for the proof of Theorem 5.1. (a) $(1, 2)$-edge, $(1, 3)$-edge, and $(1, 4)$-edge. (b) A variable gadget for a variable that occurs twice as a positive literal and once as a negative literal. Changing the flip of the $(1, 3)$-edge in the middle (variable edge) forces flipping the upper two literal edges. (c) A clause gadget for a clause of size 3.

denote the faces bounded solely by C^+ and C^- by f^+ and f^-, respectively. Without loss of generality, we assume that the gadget is embedded so that f^+ and f^- are inner faces, and we denote the outer face by f_0. The gadget represents truth values as follows. A literal edge represents the truth value `true` if and only if its path of length 1 is incident to the outer face. A variable edge represents value `true` if and only if its path of length 1 is incident to f^+. If the variable edge represents value true, then f^- is incident to a path of length 3 of the variable edge. Hence, all negative literal edges must transmit value `false`. A symmetric argument shows that if the variable edge encodes value `false`, then all positive literal edges must transmit value `false`. On the other hand, given a truth value for variable x, choosing the flips of the variable edge and all literal edges accordingly yields an embedding where each inner face has size at most 5.

A clause gadget for a clause of size 3 consists of a cycle of three $(1, 2)$-edges that correspond bijectively to the literals occurring in it; see Fig. 5.1c. Similarly, a clause of size 2 consists of a cycle on four edges, two of which are $(1, 2)$-edges corresponding to the literals. The encoding is such that a literal edge has its path of length 2 incident to the inner face of the clause gadget if and only if such a literal has value `false`. Clearly, the inner face has size at most 5 if and only if at most two literals transmit value `false`, otherwise the size is 6. Thus, the inner face of the clause gadget has size at most 5 if and only if at least one the literals transmit value `true`.

We now construct a graph G_φ as follows. We create for each variable a corresponding variable gadget and for each clause a corresponding clause gadget. We then identify literal edges of variables and clauses that correspond to the same literal. By adhering to the planar embedding of the variable–clause graph of φ, the resulting graph G_φ is planar and can be embedded such that all inner faces of the gadgets are faces of the graph. Denote this plane graph by H_φ. To obtain G_φ, we arbitrarily triangulate all faces of H_φ that are not internal faces of a gadget. Then, the only embedding choices of G_φ are the flips of the $(1,2)$- and $(1,3)$-edges. We claim that G_φ admits an embedding where every face has size at most 5 if and only if φ is satisfiable.

If G is satisfiable, pick a satisfying truth assignment. We flip each variable edge and each literal edge to encode its truth value in the assignment. As argued above, every inner face of a variable now has size at most 5, and, since each clause contains at least one satisfied literal, also the inner faces of the clause gadgets have size at most 5. Conversely, given a planar embedding of G_φ where each face has size at most 5, we construct a satisfying truth assignment for φ by assigning a variable the truth value encoded by the variable edge in the corresponding gadget. Due to the above properties, it follows that all edges corresponding to a negative literal must contribute a path of length 2 to each clause gadget containing such a literal. However, each inner face of a clause gadget has only size 5, and hence at least one of the literal edges must contribute a path of only length 1, i.e., the clause contains a satisfied literal. Since the construction of G_φ can clearly be done in polynomial time, this finishes the proof for $k = 5$.

For $k > 5$, it suffices to lengthen all cycles of the construction by $k - 5$ edges. All arguments naturally carry over. □

Polynomial-Time Algorithm for Small Faces

Next, we show that k-MINMAXFACE is polynomial-time solvable for $k = 3, 4$. Note that, if the input graph is simple, the problem for $k = 3$ is solvable if and only if the input graph is maximal planar. A bit more work is necessary if we allow parallel edges.

Let G be a biconnected planar graph. We devise a dynamic program on the SPQR-tree $\mathcal{T}$ of G. Let $\mathcal{T}$ be rooted at an arbitrary Q-node and let μ be a node of $\mathcal{T}$. We call the clockwise and counterclockwise paths connecting the poles of μ along the outer face the *boundary paths* of $pert(\mu)$. We say that an embedding of $pert(\mu)$ has type (a, b) if and only if all its inner faces have size at most k and its boundary paths have length a and b, respectively. Such an embedding is also called an (a, b)-embedding. We assume that $a \leq b$.

Clearly, each of the two boundary paths of $pert(\mu)$ in an embedding $\mathcal{E}_\mu$ of type (a, b) will be a proper subpath of the boundary of a face in any embedding of G where the embedding of $pert(\mu)$ is $\mathcal{E}_\mu$. Hence, when seeking an embedding where all faces have size at most k, we are only interested in the embedding $\mathcal{E}_\mu$ if $1 \leq a \leq b \leq k-1$. We define a partial order on the embedding types by $(a', b') \preceq (a, b)$ if and only if $a' \leq a$ and $b' \leq b$. Replacing an (a, b)-embedding $\mathcal{E}_\mu$ of $pert(\mu)$ by (a reflection of) an (a', b')-embedding $\mathcal{E}'_\mu$ with $(a', b') \preceq (a, b)$ does not create faces of size more than k; all inner faces of $\mathcal{E}'_\mu$ have size at most k by assumption, and the only other faces affected are the two faces incident to the two boundary paths of $\mathcal{E}'_\mu$, whose length does not increase. We thus seek to compute for each node μ the minimal pairs (a, b) for which it admits an (a, b)-embedding. We remark that $pert(\mu)$ can admit an embedding of type $(1, b)$ for any value of b only if μ is either a P-node or a Q-node.

We now present the algorithm for $k = 3$, which works even if we allow parallel edges.

Theorem 5.2 *3-*MINMAXFACE *can be solved in linear time for biconnected graphs.*

Proof: Clearly, the only interesting types of embeddings are $(1, 1), (1, 2)$ and $(2, 2)$ and $\preceq$ defines a total ordering on them. We thus seek to determine for each pertinent graph bottom-up in the SPQR-tree the smallest type (with respect to $\preceq$) of a valid planar embedding. For Q-nodes this is $(1, 1)$. Now consider an R-node or S-node μ. By the above remark its only possible type of embedding can be $(2, 2)$. Since every face is bounded by at least three edges, it is not hard to see that $pert(\mu)$ admits a $(2, 2)$-embedding if and only if every face of $sk(\mu)$ has size 3 and all children admit $(1, 1)$-embeddings.

For a P-node, we observe that none of its children can have a $(1, 2)$-embedding, as no two P-node can be adjacent. Thus, all children admit either a $(1, 1)$-embedding, then they are Q-nodes, or they admit a $(2, 2)$-embedding. We denote the virtual edges in $sk(\mu)$ by $(1, 1)$-edges and $(2, 2)$-edges, respectively, according to the type of embedding the corresponding graph admits. To obtain an embedding where all faces have size at most 3, we have to choose the embedding of $sk(\mu)$ in such a way that every $(2, 2)$-edge is adjacent to either two $(1, 1)$-edges or to a $(1, 1)$-edge and the parent edge. Let a and b denote the number of $(1, 1)$-edges and $(2, 2)$-edges in $sk(\mu)$, respectively. Clearly, an ordering satisfying these requirements exists if and only if $a \geq b - 1$; otherwise we necessarily have two adjacent $(2, 2)$-edges. To find a good sequence, we proceed as follows. If $a = b-1$, the sequence must alternatingly consist of $(2, 2)$-edges and $(1, 1)$-edges, starting with a $(1, 1)$-edge. The type of the resulting embedding is $(2, 2)$ and one cannot do better. If $a = b$, we do the same, but the type of the resulting embedding is $(1, 2)$; again one cannot do better. Finally, if $a \geq b+1$, we

again do the same, and finally append the remaining $(1,1)$-edges. Then the resulting embedding has type $(1,1)$.

Clearly, we can process each node μ in time proportional to the size of its skeleton. The graph admits an embedding if and only if the pertinent graph of the child of the root admits some valid embedding. □

We now deal with the case $k = 4$, which is similar but more complicated. The relevant types are $(1,1), (1,2), (1,3), (2,2), (2,3)$, and $(3,3)$. We note that precisely the two elements $(2,2)$ and $(1,3)$ are incomparable with respect to $\preceq$. Thus, it seems that, rather than computing only the single smallest type for which each pertinent graph admits an embedding, we are now forced to find all minimum pairs for which the pertinent graph admits a corresponding embedding. However, by the above observation, if a pertinent graph $pert(\mu)$ admits a $(1,3)$-embedding, then μ must be a P-node. However, if the parent of μ is an S-node or an R-node, then using a $(1,3)$-embedding results in a face of size at least 5. Thus, such an embedding can only be used if the parent is the root Q-node. If there is the choice of a $(2,2)$-embedding in this case, it can of course also be used at the root. Therefore, we can mostly ignore the $(1,3)$-case and consider the linearly ordered embedding types $(1,1), (1,2), (2,2), (2,3)$ and $(3,3)$. The type $(1,3)$ is only relevant for P-nodes whose pertinent graph admits an embedding of type $(1,3)$ embedding but no embedding of type $(2,2)$.

Theorem 5.3 *4-*MINMAXFACE *can be solved in $O(n^{1.5})$ time for biconnected graphs.*

Proof: We process the SPQR-tree of the input graph in a bottom-up fashion. The pertinent graphs of Q-nodes admit embeddings of type $(1,1)$.

Now consider an S- or an R-node μ. All faces of $sk(\mu)$ must have size at most 4. Moreover, since all faces have length at least 3, a valid embedding of $pert(\mu)$ does not exist if some child only allows embeddings of type $(1,3)$, $(2,3)$ or $(3,3)$. Thus, the only freedom is to choose the flips of the pertinent graphs admitting only $(1,2)$-embeddings. A face can receive only a single path of length 2 from one of its incident edges, and this is possible only if the face is a triangle and none of its incident edges is a $(2,2)$-edge. We thus seek a matching between the $(1,2)$-edges and their incident faces that can receive a path of length 2. Depending on the size of the faces incident to the parent edge and whether they need to receive a path of length 2 in order to find a valid embedding, the type is either $(2,2)$ (if both faces are triangles and they do not need to receive a path of length 2), $(2,3)$ (if one is a triangle that does not need to receive a path of length 2) or $(3,3)$ (remaining cases).

Now consider a P-node. Each child must have an embedding of type $(1,1)$, $(2,2)$, $(2,3)$ or $(3,3)$. Again, we denote the edges whose corresponding pertinent graph admits an embedding of type (a,b) as (a,b)-edges.

First observe that removing in any embedding all $(2,2)$-edges except for one and placing them next to the single $(2,2)$-edge we did not remove results in a valid embedding whose boundary paths do not increase. Thus, we can assume without loss of generality that there is at most one $(2,2)$-edge. Moreover, if there is a $(2,3)$-edge, we can actually move the $(2,2)$-edge next to it without increasing the size of any face. Thus, if there are any $(2,3)$-edges we can even assume that there is no $(2,2)$-edge.

Let us first assume that there is no $(2,3)$-edge. We then have to choose the embedding such that $(1,1)$-edges alternate with $(3,3)$-edges and the single $(2,2)$-edge. We append any excess of $(1,1)$-edges at the end. Let a denote the number of $(1,1)$-edges and let b denote the total number of $(2,2)$- and $(3,3)$-edges. A valid embedding exists only if $a \geq b-1$. In this case a suitable sequence always exists. If possible, we start and end with a $(1,1)$-edge, resulting in a $(1,1)$-embedding. If this is not the case, we try to start with a $(1,1)$ and put the $(2,2)$ in the end if it exists. Then we obtain a $(1,2)$-embedding if there is a $(2,2)$-edge and a $(1,3)$-embedding otherwise. If this is also not possible since $a = b-1$, we start with the $(2,2)$-edge if it exists. This results in either a $(2,3)$ or a $(3,3)$-embedding.

Assume now that there exist $(2,3)$-edges. We first observe that recursively merging pairs of $(2,3)$-edges into single $(3,3)$-edges by placing them adjacent in the skeleton of the P-node and by flipping their embeddings in such a way that their sides of length 2 are incident to the same face, we did not remove results in a valid embedding whose boundary paths do not increase. Thus, we can assume without loss of generality that there is at most one $(2,3)$-edge. As already observed, we can also assume that the remaining edges are only $(1,1)$-edges and $(3,3)$-edges. We then have to choose the embedding such that $(1,1)$-edges alternate with $(3,3)$-edges and the single $(2,3)$-edge. We append any excess of $(1,1)$-edges at the end. Let a denote the number of $(1,1)$-edges and let b denote the total number of $(2,3)$- and $(3,3)$-edges. A valid embedding exists only if $a \geq b-1$. In this case a suitable sequence always exists. If possible, we start and end with a $(1,1)$-edge, resulting in a $(1,1)$-embedding. If this is not the case, we try to start with a $(1,1)$ and put the $(2,3)$-edge in the end, thus obtaining a $(1,2)$-embedding. If this is also not possible since $a = b-1$, we start with the $(2,3)$-edge. This results in a $(2,3)$-embedding.

The bottleneck concerning the running time is finding the matching for treating the R-node, which can be solved in $O(n^{1.5})$ time [Gab83]. □

Approximation Algorithm

In this section, we present a constant-factor approximation algorithm for the problem of minimizing the largest face in an embedding of a biconnected graph $G = (V, E)$.

We again solve the problem by dynamic programming on the SPQR-tree of G.

Let G be a biconnected planar graph, and let $\mathcal{T}$ be its SPQR-tree, rooted at an arbitrary Q-node. Let μ be a node of $\mathcal{T}$. We shall consider the embeddings of $pert(\mu)$ where the two poles are embedded on the outer face. We also include the parent edge in the embedding, by drawing it in the outer face. In such an embedding of $sk(\mu)$, the two faces incident to the parent edge are called *the outer faces*, while the remaining faces are *inner faces*.

Recall that an (a, b)-*embedding* of $pert(\mu)$ is an embedding whose boundary paths have lengths a and b, where we always assume that $a \leq b$. We say that an (a, b)-embedding of $pert(\mu)$ is *out-minimal* if for any (a', b')-embedding of $pert(\mu)$, we have $a \leq a'$ and $b \leq b'$. Note that an out-minimal embedding need not exist; e.g., $pert(\mu)$ may admit a $(2, 4)$-embedding and a $(3, 3)$-embedding, but no (a, b)-embedding with $a \leq 2$ and $b \leq 3$. We will later show, however, that such a situation can only occur when μ is an S-node.

Let $\textsc{Opt}(G)$ denote the smallest integer k such that G has an embedding whose every face has size at most k. For a node μ of $\mathcal{T}$, we say that an embedding of $pert(\mu)$ is *c-approximate*, if each inner face of the embedding has size at most $c \cdot \textsc{Opt}(G)$.

Call an embedding of $pert(\mu)$ *neat* if it is out-minimal and 6-approximate. The main result of this section is the next proposition.

Proposition 1 *Let G be a biconnected planar graph with SPQR tree $\mathcal{T}$, rooted at an arbitrary Q-node. Then the pertinent graph of every Q-node, P-node or R-node of $\mathcal{T}$ has a neat embedding, and this embedding may be computed in polynomial time.*

Since the pertinent graph of the root of $\mathcal{T}$ is the whole graph G, the proposition implies a polynomial 6-approximation algorithm for minimization of largest face.

Our proof of Proposition 1 is constructive. Fix a node μ of $\mathcal{T}$ which is not an S-node. We now describe an algorithm that computes a neat embedding of $pert(\mu)$, assuming that neat embeddings are available for the pertinent graphs of all the descendant nodes of μ that are not S-nodes. We distinguish cases based on the type of the node μ.

Non-root Q-nodes. As a base case, suppose that μ is a non-root Q-node of $\mathcal{T}$. Then $pert(\mu)$ is a single edge, and its unique embedding is clearly neat.

P-nodes. Next, suppose that μ is a P-node with k child nodes $\mu_1, \ldots, \mu_k$, represented by k skeleton edges $e_1, \ldots, e_k$. Let G_i be the expansion graph of e_i. We construct the *expanded skeleton* $sk^*(\mu)$ as follows: if for some i the child node μ_i is

an S-node whose skeleton is a path of length m, replace the edge e_i by a path of length m, whose edges correspond in a natural way to the edges of $sk(\mu_i)$.

Every edge e' of the expanded skeleton corresponds to a node μ' of $\mathcal{T}$ which is a child or a grand-child of μ. Moreover, μ' is not an S-node, and we may thus assume that we have already computed a neat embedding for $pert(\mu')$. Note that $pert(\mu')$ is the expansion graph of e'.

For each $i \in \{1, \ldots, k\}$ define ℓ_i to be the smallest value such that G_i has an embedding with boundary path of length ℓ_i. We compute ℓ_i as follows: if μ_i is not and S-node, then we already know a neat (a_i, b_i)-embedding of G_i, and we may put $\ell_i = a_i$. If, on the other hand, μ_i is an S-node, then let m be the number of edges in the path $sk(\mu_i)$, and let $G_i^1, G_i^2, \ldots, G_i^m$ be the expansion graphs of the edges of the path. For each G_i^j, we have already computed a neat (a_j, b_j)-embedding, so we may now put $\ell_i = \sum_{j=1}^{m} a_j$. Clearly, this value of ℓ_i corresponds to the definition given above.

We now fix two distinct indices $\alpha, \beta \in \{1, \ldots, k\}$, so that the values ℓ_α and ℓ_β are as small as possible; formally, $\ell_\alpha = \min\{\ell_i; i = 1, \ldots, k\}$ and $\ell_\beta = \min\{\ell_i; i = 1, \ldots, k \text{ and } i \neq \alpha\}$.

Let us fix an embedding of $sk(\mu)$ in which e_α and e_β are adjacent to the outer faces. We extend this embedding of $sk(\mu)$ into an embedding of $pert(\mu)$ by replacing each edge of $sk^*(\mu)$ by a neat embedding of its expansion graph, in such a way that the two boundary paths have lengths ℓ_α and ℓ_β. Let $\mathcal{E}$ be the resulting $(\ell_\alpha, \ell_\beta)$-embedding of $pert(\mu)$.

We now show that $\mathcal{E}$ is neat. From the definitions of ℓ_α and ℓ_β, we easily see that $\mathcal{E}$ is out-minimal. It remains to show that it is 6-approximate. Let f be any inner face of $\mathcal{E}$. If f is an inner face of the expansion graph G_i of some e_i, then f is an inner face of some previously constructed neat embedding, hence $|f| \leq 6 \cdot \textsc{Opt}(G)$.

Suppose then that f is not the inner face of any G_i. Then the boundary of f intersects two distinct expansion graphs G_i and G_j. Hence the boundary of f is the union of two paths P_i and P_j, with $P_i \subseteq G_i$ and $P_j \subseteq G_j$. Let d_i and d_j be the lengths of P_i and P_j, respectively, and assume that $d_i \leq d_j$. It follows that $|f| = d_i + d_j \leq 2d_j$. We claim that every embedding of G has a face of size at least $d_j/2$. If μ_j is not an S-node, this follows from the fact that P_j is a boundary path in an out-minimal embedding of G_j, hence any other embedding of G_j must have a boundary path of length at least d_j. If, on the other hand, μ_j is an S-node, then in every embedding of G_j, the two boundary paths have total length at least d_j, so every embedding of G_j has a boundary path of length at least $d_j/2$ and thus G has a face of size at least $d_j/2$. We conclude that $|f| \leq 2d_j \leq 4 \cdot \textsc{Opt}(G)$, showing that $\mathcal{E}$ is indeed neat.

R-nodes. Suppose now that μ is an R-node. As with P-nodes, we define the *expanded skeleton* $sk*(\mu)$ by replacing each edge of $sk(\mu)$ corresponding to an S-node by a path of appropriate length. The graph $sk*(\mu)$ together with the parent edge forms a subdivision of a 3-connected graph. In particular, its embedding is determined uniquely up to a flip and a choice of outer face. Fix an embedding of $sk^*(\mu)$ and the parent edge, so that the parent edge is on the outer face. Let f_1 and f_2 be the two faces incident to the parent edge of μ.

Let e be an edge of $sk^*(\mu)$, let G_e be its expansion graph, and let $\mathcal{E}_e$ be a neat (a, b)-embedding of G_e, for some $a \leq b$. The boundary path of $\mathcal{E}_e$ of length a will be called *the short side* of $\mathcal{E}_e$, while the boundary path of length b will be *the long side*. If $a = b$, we choose the long side and short side arbitrarily.

Our goal is to extend the embedding of $sk^*(\mu)$ into an embedding of $pert(\mu)$ by replacing each edge e of $sk^*(\mu)$ with a copy of $\mathcal{E}_e$. In doing so, we have to choose which of the two faces incident to e will be adjacent to the short side of $\mathcal{E}_e$.

First of all, if e is an edge of $sk^*(\mu)$ incident to one of the outer faces f_1 or f_2, we embed $\mathcal{E}_e$ in such a way that its short side is adjacent to the outer face. Since f_1 and f_2 do not share an edge in $sk^*(\mu)$, such an embedding is always possible, and guarantees that the resulting embedding of $pert(\mu)$ will be out-minimal.

It remains to determine the orientation of $\mathcal{E}_e$ for the edges e that are not incident to the outer faces, in such a way that the largest face of the resulting embedding will be as small as possible. Rather than solving this task optimally, we formulate a linear programming relaxation, and then apply a rounding step which will guarantee a constant factor approximation.

Intuitively, the linear program works as follows: given an edge e incident to a pair of faces f and g, and a corresponding graph G_e with a short side of length a and a long side of length b, rather than assigning the short side to one face and the long side to the other, we assign to each of the two faces a fractional value in the interval $[a, b]$, so that the two values assigned by e to f and g have sum $a + b$, and the maximum total amount assigned to a single face of $sk^*(\mu)$ from its incident edges is as small as possible.

More precisely, we consider the linear program with the set of variables

$$\{M\} \cup \{x_{e,f}; e \text{ is an edge adjacent to face } f\},$$

where the goal is to minimize M subject to the following constraints:

- For every edge e adjacent to a pair of faces f and g, we have the constraints $x_{e,f} + x_{e,g} = a + b$, $a \leq x_{e,f} \leq b$ and $a \leq x_{e,g} \leq b$, where $a \leq b$ are the lengths of the two boundary paths of $\mathcal{E}_e$.

- Moreover, if an edge e is adjacent to an outer face $f \in \{f_1, f_2\}$ as well as an inner face g, then we set $x_{e,f} = a$ and $x_{e,g} = b$, with a and b as above.
- For every inner face f of $sk^*(\mu)$, we have the constraint $\sum_e x_{e,f} \leq M$, where the sum is over all edges incident to f.

Given an optimal solution of the above linear program, we determine the embedding of $pert(\mu)$ as follows: for an edge e of $sk^*(\mu)$ incident to two inner faces f and g, if $x_{e,f} \leq x_{e,g}$, embed $\mathcal{E}_e$ with its short side incident to f and long side incident to g. Let $\mathcal{E}_\mu$ be the resulting embedding.

We claim that $\mathcal{E}_\mu$ is neat. We have already seen that $\mathcal{E}_\mu$ is out-minimal, so it remains to show that every inner face of $\mathcal{E}_\mu$ has size at most $6 \cdot \textsc{Opt}(G)$. Let us say that an inner face of $\mathcal{E}_\mu$ is *deep* if it is also an inner face of some $\mathcal{E}_e$, and it is *shallow* if it corresponds to a face of $sk^*(\mu)$. Note that the deep faces have size at most $6 \cdot \textsc{Opt}(G)$, since all the $\mathcal{E}_e$ are neat embeddings, so we only need to estimate the size of the shallow faces.

Let $\textsc{Opt}_{out}$ denote the minimum k such that $pert(\mu)$ has an out-minimal embedding whose every shallow face has size at most k. We claim that $\textsc{Opt}_{out} \leq 3 \cdot \textsc{Opt}(G)$. To see this, consider an embedding of $pert(\mu)$ in which each face has size at most $\textsc{Opt}(G)$. In this embedding, replace each subembedding of G_e by a copy of $\mathcal{E}_e$, without increasing the size of any shallow face. This can be done, because each $\mathcal{E}_e$ is out-minimal. Call the resulting embedding $\mathcal{E}'$. Next, for every edge e of $sk^*(\mu)$ adjacent to f_1 or f_2, flip $\mathcal{E}_e$ so that its short side is incident to f_1 or f_2. Let $\mathcal{E}''$ be the resulting embedding of $pert(\mu)$. Clearly, $\mathcal{E}''$ is out-minimal.

In $\mathcal{E}''$, some inner shallow face f adjacent to f_1 or f_2 may have larger size than the corresponding face of $\mathcal{E}'$; however, for such an f, its size in $\mathcal{E}''$ is at most equal to the sum of the sizes of f, f_1 and f_2 in $\mathcal{E}'$. In particular, each inner shallow face has size at most $3 \cdot \textsc{Opt}(G)$ in $\mathcal{E}''$, and hence $\textsc{Opt}_{out} \leq 3 \cdot \textsc{Opt}(G)$, as claimed.

We will now show that each shallow face of $\mathcal{E}_\mu$ has size at most $2 \cdot \textsc{Opt}_{out}$. Let M be the value of optimum solution to the linear program defined above. Clearly, $M \leq \textsc{Opt}_{out}$, since from an out-minimal embedding with shallow faces of size at most $\textsc{Opt}_{out}$, we may directly construct a feasible solution of the linear program with value $\textsc{Opt}_{out}$. Let f be a shallow face of $\mathcal{E}_\mu$. Let e be an edge of $sk^*(\mu)$ incident to f, and let g be the other face incident to e. Let a and b be the lengths of the short side and long side of $\mathcal{E}_e$, respectively. If $x_{e,f} \leq x_{e,g}$, then $\mathcal{E}_e$ contributes to the boundary of f by its short side, which has length a. Otherwise, f has the long side of $\mathcal{E}_e$ on its boundary, but that may only happen when $x_{e,f} \geq x_{e,g}$, and hence $b \leq a + b = x_{e,f} + x_{e,g} \leq 2x_{e,f}$. From this, we see that f has size at most $\sum_e 2x_{e,f} \leq 2M$, with the previous sum ranging over all edges of $sk^*(\mu)$ incident to f.

Thus, for every shallow face f of $\mathcal{E}_\mu$, we have $|f| \leq 2M \leq 2 \cdot \text{OPT}_{out} \leq 6 \cdot \text{OPT}(G)$, showing that $\mathcal{E}_\mu$ is neat.

The root Q-node. Finally, suppose that μ is the root of the SPQR-tree $\mathcal{T}$. That means that μ is a Q-node, and its skeleton is formed by two parallel edges e_1 and e_2, where the expansion graph of e_1 is a single edge and the expansion graph G_2 of e_2 is the pertinent graph of the unique child node μ' of μ. If μ' is not an S-node, we already have a neat (a, b)-embedding $\mathcal{E}_2$ of G_2, and by inserting the edge e_1 to this embedding in such a way that the outer face has size $a + 1$, we clearly obtain a neat embedding of G. If μ' is an S-node, then G_2 is a chain of biconnected graphs $G_2^1, G_2^2, \ldots, G_2^k$, and for each G_2^i we have a neat (a_i, b_i)-embedding. Combining these embedding in an obvious way, and adding the edge e_1, we get an embedding of G whose outer face has size $1 + a_1 + a_2 + \cdots + a_k$, and whose unique inner shallow face has size $1+b_1+b_2+\cdots+b_k$. Since in each embedding of G, the two faces incident to e_1 have total size at least $2 + a_1 + \cdots + a_k + b_1 + \cdots + b_k$, we conclude that our embedding of G is neat.

This completes the proof of Proposition 1, and yields a 6-approximation algorithm for the minimization of largest face in biconnected graphs.

Theorem 5.4 *A 6-approximation for* MINMAXFACE *in biconnected graphs can be computed in polynomial time.*

5.4 Perfectly Uniform Face Sizes

In this section we study the problem of deciding whether a biconnected planar graph admits a k-uniform embedding. Note that, due to Euler's formula, a connected planar graph with n vertices and m edges has $f = m - n + 2$ faces. In order to admit an embedding where every face has size k, it is necessary that $2m = fk$. Hence there is at most one value of k for which the graph may admit a k-uniform embedding.

In the following, we characterize the graphs admitting 3-uniform and 4-uniform embeddings, and we give an efficient algorithm for testing whether a graph admits a 6-uniform embedding. Finally, we show that testing whether a graph admits a k-uniform embedding is NP-complete for odd $k \geq 7$ and even $k \geq 10$. We leave open the cases $k = 5$ and $k = 8$.

Our characterizations and our testing algorithm use the recursive structure of the SPQR-tree. To this end, it is necessary to consider embeddings of pertinent graphs, where we only require that the interior faces have size k, whereas the outer face may have different size, although it must not be too large. We call such an embedding

almost k-uniform. The following lemma states that the size of the outer face in such an embedding depends only on the number of vertices and edges in the pertinent graph.

Lemma 5.1 *Let G be a graph with n vertices and m edges with an almost k-uniform embedding. Then the outer face has length $\ell = k(n - m - 1) + 2m$.*

Proof: Let f denote the number of faces of G in a planar embedding, which us uniquely determined by Euler's formula $n - m + f = 2$. By double counting, we find that $(f - 1) \cdot k + \ell = 2m$. Euler's formula implies that $f = 2 + m - n$, and plugging this into the second formula, we obtain that $(1 + m - n) \cdot k + \ell = 2m$ or, equivalently, $\ell = k(n - m - 1) + 2m$. □

Thus, for small values of k, where the two boundary paths of the pertinent graph may have only few different lengths, the type of an almost k-uniform embedding is essentially fixed.

Characterization for $k = 3, 4$

For 3-uniform embeddings first observe that every facial cycle must be a triangle. If the input graph is simple, then this implies that it must be a triangulation. Then the graph is 3-connected and the planar embedding is uniquely determined. We characterize the multi-graphs that have such an embedding.

Theorem 5.5 *A biconnected planar graph G admits 3-uniform embedding if and only if its SPQR-tree satisfies all of the following conditions.*

(i) S- and R-nodes are only adjacent to Q- and P-nodes.
(ii) Every R-node skeleton is a planar triangulation.
(iii) Every S-node skeleton has size 3.
(iv) Every P-node with k neighbors has k even and precisely $k/2$ of the neighbors are Q-nodes.

Proof: It is not hard to see that all conditions are necessary. We prove sufficiency. To this end, we choose the embeddings of the R-node skeletons arbitrarily, and we embed the P-node skeletons such that virtual edges corresponding to Q-node and non-Q-node neighbors alternate. We claim that in the resulting planar embedding of G all faces have size 3.

To this end, root the SPQR-tree $\mathcal{T}$ of G at an arbitrary edge e and consider the embedding e incident to the outer face.

Claim 5.1 *1. If μ is a Q-node or a P-node whose parent is not a Q-node, then it has an almost 3-uniform embedding of type $(1, 1)$.*
2. If μ is an S-node, an R-node, or a P-node whose parent is a Q-node, then it has an almost 3-uniform embedding of type $(2, 2)$.

We prove this by induction on the height of the node in the SPQR-tree. Clearly, the statement holds for Q-nodes. Now consider an internal node μ and assume that the claim holds for all children.

If μ is an S-node, then the clockwise (counterclockwise) path of $pert(\mu)$ between the poles along the outer face is the concatenation of the clockwise path (counterclockwise) paths of the pertinent graphs of its children. By property (iii) there are only two children and by property (i) they are either Q- or P-nodes. By the inductive hypothesis, their embeddings are almost 3-uniform and have type $(1, 1)$, and hence the type of the embedding of $pert(\mu)$ is $(2, 2)$.

If μ is an R-node, then its clockwise (counterclockwise) path between the poles is the concatenation of the clockwise (counterclockwise) paths of the pertinent graphs corresponding to the edges on the clockwise (counterclockwise) path between the poles. By property (ii) each of these paths has length 2 in $sk(\mu)$ and the children are either Q- or P-nodes. Thus, by the inductive hypothesis, their embeddings have type $(1, 1)$.

If μ is a P-node whose parent is not a Q-node, then, by our choice of the planar embedding, the two outer paths in $sk(\mu)$ are edges corresponding to Q-nodes, and the claim follows from the inductive hypothesis. If the parent of μ is a Q-node, then, again by the embedding choice, the two edges outer paths in $sk(\mu)$ are edges corresponding to S- or R-nodes, and again the inductive hypothesis implies the claim. This finishes the proof of the claim.

Let now μ denote the Q-node corresponding to the root edge e and consider the two faces incident to e, which show up as faces in $sk(\mu)$. Let μ' be the neighbor of μ in the SPQR-tree. Then μ' is either an S-node, an R-node, or a P-node whose parent is a Q-node. In all cases the embedding of $pert(\mu')$ has type $(2, 2)$, and hence the two faces incident to e have size 3. Since e was chosen arbitrarily, it follows that each face has size 3. □

Corollary 5.1 *It can be tested in linear time whether a biconnected planar graph admits a 3-regular dual.*

For 4-uniform embeddings observe that every facial cycle must be a simple cycle of length 4. Since every planar graph containing a cycle of odd length also has a face of odd length in any planar embedding, it follows that the graph must be bipartite.

Now, if the graph is simple, the graph must be planar, bipartite and each face must have size 4. It is well known (and follows from Euler's formula) that this is the case if and only if the graph has $2n-4$ edges; the maximum number of edges for a simple bipartite planar graph. Again, if the graph is not simple more work is necessary. For a virtual edge e in a skeleton $sk(\mu)$, we denote by m_e and n_e the number of edges in its expansion graph.

Theorem 5.6 *A biconnected planar graph admits a 4-regular dual if and only if it is bipartite and satisfies the following conditions.*

(i) For each P-node either all expansion graphs satisfy $m_e = 2n_e - 4$, or half of them satisfy $m_e = 2n_e - 5$ and the other half are Q-nodes.

(ii) For each S- or R-node all faces have size 3 or 4; the expansion graphs of all edges incident to faces of size 4 satisfy $m_e = 2n_e - 3$ and for each triangular face, there is precisely one edge whose expansion graph satisfies $m_e = 2n_e - 4$, the others satisfy $m_e = 2n_e - 3$.

Proof: We choose the planar embedding as follows. For each P-node, if half of the neighbors are Q-nodes, then we choose the embedding such that Q-nodes and non-Q-nodes alternate. All remaining embedding choices can be done arbitrarily. We claim that in the resulting embedding all faces have size 4.

As in the proof of Theorem 5.5, root the SPQR-tree $\mathcal{T}$ of G at an arbitrary edge e and consider the embedding as having e incident to the outer face.

Claim 5.2 *For each node μ of $\mathcal{T}$ in the embedding of $pert(\mu)$ without the parent edge denote by ℓ_μ and r_μ the length of the clockwise and counterclockwise path on the outer face connecting the poles of μ.*

1. *Each internal face of $pert(\mu)$ has size 4.*
2. *If μ is a Q-node or a P-node with Q-node neighbors whose parent is not a Q-node, then $pert(\mu)$ has an almost 4-uniform embedding of type $(1,1)$.*
3. *If μ is a P-node whose neighbors all satisfy $m_e = 2n_e - 4$, or μ is an S- or an R-node whose parent satisfies $m_e = 2n_e - 4$, then $pert(\mu)$ has an almost 4-uniform embedding of type $(2,2)$.*
4. *If μ is a P-node with Q-node neighbors whose parent is a Q-node, or if μ is an S- or an R-node whose parent is a Q-node or satisfies $m_e = 2n_e - 3$, then $pert(\mu)$ has an almost 4-uniform embedding of type $(3,3)$.*

The proof of the claim is by structural induction on the SPQR-tree. Clearly, it holds for the leaves, which are Q-nodes. Now consider an internal node μ.

If μ is a P-node, with Q-node neighbors whose parent is not a Q-node then, by property (i) all children have almost 4-uniform embeddings. Further, the children that

are not Q-nodes satisfy $m_e = 2n_e - 4$, and hence, their outer face has size 6 by Lemma 5.1. It must hence be an S- or an R-node and, by the inductive hypothesis, their embeddings have type $(3, 3)$. Thus, the alternation of Q-nodes and these children ensures that inner faces have size 4. Moreover, since the parent is not a Q-node, the linear ordering of the children (excluding the parent) starts and ends with a Q-node. Hence the embedding of $pert(\mu)$ has type $(1, 1)$.

If μ is a P-node whose neighbors all satisfy $m_e = 2n_e - 4$, then all children have almost 4-uniform embeddings whose outer faces have size 4 by Lemma 5.1. Since a non-P-node cannot have an embedding of type $(1, x)$ for any value of x, their embeddings have type $(2, 2)$. This implies the inductive hypothesis.

If μ is a P-node with Q-node neighbors whose parent is a Q-node, then one more than half of its children satisfy $m_e = 2n_e - 4$ and hence have almost 4-uniform embeddings of type $(3, 3)$. The alternation of Q-nodes and these children implies the statement.

If μ is an S- or an R-node whose parent satisfies $m_e = 2n_e - 3$ (or it is a Q-node), the internal faces have size 4 according to the inductive hypothesis and property (ii). A similar argument shows that the embedding has type $(3, 3)$.

If μ is an S- or an R-node whose parent satisfies $m_e = 2n_e - 4$, then the two faces incident to the parent edge are triangles, and the children all satisfy $m_e = 2n_e - 3$, and hence have almost 4-uniform embeddings of type $(1, 1)$. Thus the embedding of $pert(\mu)$ has type $(2, 2)$. This finishes the proof of the claim, and as it immediately implies that every face has size 4, also the proof of the theorem. □

Corollary 5.2 *It can be tested in linear time whether a biconnected planar graph admits a 4-regular dual.*

Testing Algorithm for 6-Uniform Embeddings

To test the existence of a 6-uniform embedding, we again use bottom-up traversal of the SPQR-tree and are therefore interested in the types of almost 6-uniform embeddings of pertinent graphs. Clearly, each of the two boundary paths of a pertinent graph, may have length at most 5. Thus, only embedding of type (a, b) with $1 \leq a \leq b \leq 5$ are relevant. Although, by Lemma 5.1 the value of $a + b$ is fixed, this does usually not uniquely determine the values a and b in this case. For example, at first sight it may seem that if the outer face of a pertinent graph has length 6, then uniform embeddings of type $(1, 5)$, $(2, 4)$ and $(3, 3)$ may all be possible. However, as we will argue in the following, only one of these choices is relevant in any situation.

In order to admit a k-uniform embedding with k even, it is necessary that the graph is bipartite. In particular, this implies that also the outer face of any pertinent graph

must have even length. For a 6-uniform embedding the length of the face must be in $\{2, 4, 6, 8, 10\}$. Let us now investigate for each such length the possible types of almost 6-uniform embeddings.

For length 2 and length 10, the types must be $(1, 1)$ and $(5, 5)$, respectively. For length 4, the type must be $(1, 3)$ or $(2, 2)$. However, the poles of $sk(\mu)$ are either in the same color class of the bipartite graph of G, then only $(2, 2)$ is possible, or they belong to different color classes, then only $(1, 3)$ is possible. For length 6, the possible types are $(1, 5), (2, 4)$ and $(3, 3)$. However, type $(1, 5)$ implies that one of the paths consists of a single edge, i.e., μ is a P-node. However, due to the path of length 5 on the other boundary, we need another parallel edge to achieve faces of size 6. However, such an edge must be a Q-node child of μ, showing that $(1, 5)$ cannot occur. Thus only $(2, 4)$ and $(3, 3)$ are actually possible. Again, the color class of the poles determines the pair uniquely. Finally, for length 8, the possible types are $(3, 5)$ and $(4, 4)$ and again the color classes uniquely determine the type.

Thus, we know for each internal node μ precisely what must be the type of an almost 6-uniform embedding of $pert(\mu)$ if one exists. It remains to check whether for each node μ, assuming that all children admit an almost 6-uniform embedding of the correct type, it is possible to put them together to an almost 6-uniform embedding of $pert(\mu)$ of the correct type. For this, we need to decide (i) an embedding of $sk(\mu)$ and (ii) for each child whether to use the to mirror its almost k-uniform embedding. We refer to the latter decision as choosing the flip of the child.

For S-nodes, which must necessarily have length at most 6, we can simply try all ways to choose the flips of the children and see whether one of them gives the correct values.

For a P-node, observe that, in order to obtain an almost 6-uniform embedding, the boundary paths of the children must be either all odd or all even. If they are all even, then all pertinent graphs of children must have types $(2, 2)$, $(2, 4)$ or $(4, 4)$. Clearly, the children with types $(2, 2)$ and $(4, 4)$ have to alternate in the sequence, the children of type $(2, 4)$ can be inserted at an arbitrary place. Let a and b denote the number of children of type $(2, 2)$ and $(4, 4)$, respectively. It is necessary that $|a - b| = 1$, otherwise they cannot alternate. If $a > b$, then the type of $pert(\mu)$ must be $(2, 2)$, if $b < a$, it must be $(4, 4)$ and if $a = b$, then it must be $(2, 4)$.

The case that all paths are odd is similar but slightly more complicated as there are more possible embedding types for the children. The possible types are $(1, 1)$, $(3, 3)$, $(3, 5)$, and $(5, 5)$ (recall that $(1, 3)$ and $(1, 5)$ cannot occur in a P-node). Again, we call the corresponding virtual edges $(1, 1)$-, $(3, 3)$-, $(3, 5)$- and $(5, 5)$-edges, respectively.

We now perform some simple groupings of such virtual edges that can be assumed to be placed consecutively in any valid embedding. We view these consecutive edges as a single child whose outer boundary paths determine its type of embedding. First,

observe that if there is no $(3,5)$-edge but a $(3,3)$-edge, then all children must necessarily be of type $(3,3)$. In this case any embedding of $sk(\mu)$ works and yields an embedding of type $(3,3)$ for $pert(\mu)$. Otherwise, we group the $(3,5)$-edges together with all $(3,3)$-edges into one big chunk, which then represents a child of type $(3,5)$. Thus, we can assume that no $(3,3)$-edge exists. Next, observe that the $(3,5)$-edges must occur in pairs whose interior face is bounded by two paths of length 3. Viewed as one graph, the type of their embedding is $(5,5)$. Note that, due to the pairing, one $(3,5)$ might be left over. In this case, we start the ordering for the embedding of $sk(\mu)$ with the $(3,5)$-edge. We then alternatingly insert $(1,1)$-edges and $(5,5)$-edges. If we manage to use up all virtual edges, we have found a valid embedding. Otherwise, since we only followed necessary conditions, a valid embedding does not exist.

For an R-node μ, observe that each face of the skeleton has size at least 3. Thus children whose almost 6-uniform embedding has type $(x,5)$ for some value of x immediately exclude the existence of a 6-uniform embedding for $pert(\mu)$. It now remains to choose the flips of the almost 6-uniform embeddings of the children. Note that for children whose type (a,b) is such that $a = b$, this choice does not matter. Thus, only the flips of children of types $(1,3)$ and $(2,4)$ matter. We initially consider each face as having a demand of 6. However, for each edge of type (a,b) incident to a face f, we remove from the demand of face f the amount $\min\{a,b\}$, and rather conceptually replace the edge by a $(0,|a-b|)$-edge. Due to the above observation, the only types of edges remaining are $(0,0)$ and $(0,2)$. Clearly, we can ignore the $(0,0)$-edges. The remaining $(0,2)$-edges can pass two units of boundary length into one of their incident faces. We now consider the demand of each face. Clearly, it is necessary that these demands are even. We then model this as a matching problem, where each $(0,2)$-edge has capacity 1, and each face has capacity half its demand. We then seek a generalized perfect matching in the incidence graph of faces and vertices with positive capacity such that each vertex is matched to exactly as many edges as its capacity. This can be solved in $O(n^{1.5})$ time by an algorithm due to Gabow [Gab83]. Clearly an embedding exists if and only the corresponding matching exists. We thus have proved the following theorem.

Theorem 5.7 *It can be tested in $O(n^{1.5})$ time whether a biconnected planar graph admits a 6-uniform embedding.*

Uniform Embeddings with Large Faces

We prove NP-hardness for testing the existence of a k-uniform embedding for $k = 7$ and $k \geq 9$ by giving a reduction from the NP-complete problem PLANAR POSITIVE 1-IN-3-SAT where each variable occurs at least twice and at most three times and

each clause has size two or three. The NP-completeness of this version of satisfiability follows from the results of Moore and Robson [MR01], as shown by the following Theorem.

Theorem 5.8 PLANAR POSITIVE 1-IN-3-SAT *is NP-complete even if each variable occurs two or three times and each clause has size two or three.*

Proof: Clearly the problem is in NP. For the hardness proof, we reduce from the NP-complete problem CUBIC PLANAR MONOTONE 1-IN-3-SAT, a variant of planar 3-SAT where each variable occurs three times and each clause consists of three literals that are either all positive or all negative [MR01]. We denote 1-in-3 clauses as (x, y, z) (or (x, y) for clauses of size two) where x, y, z are literals.

Consider a planar embedding of the variable–clause graph and a clause $C = (\neg x, y, z)$ where a variable x occurs negated. We now replace C by two clauses $C' = (x', y, z)$ and $C'' = (x', x)$, where x' is a new variable. Observe that, in the variable–clause graph this corresponds to subdividing the edge xC twice. Thus, the resulting variable–clause graph remains planar. Further, the clause C'' ensures that, in any satisfying 1-in-3 truth assignment, the variables x and x' have complementary truth values, i.e., x' is the negation of x. Thus the resulting instance of PLANAR POSITIVE 1-IN-3-SAT is equivalent to the original one. Moreover, the new instance has one fewer negated literal. After $O(n)$ such operations, we obtain an equivalent instance where all literals are positive. Obviously the resulting formula satisfies the claimed properties and the reduction can be performed in polynomial time. □

Theorem 5.9 k-UNIFORMFACES *is NP-complete for all odd $k \geq 7$.*

Proof: We reduce from PLANAR POSITIVE 1-IN-3-SAT where each variable occurs two or three times and each clause has size two or three, which is NP-complete by Theorem 5.8. Let φ be such a formula with n variables, C clauses and L literals (total number of literals in all clauses), and let G_φ be its variable–clause graph embedded in the plane. We add an additional vertex s, which we call *sink* into the outer face and connect each variable to the sink in such a way that no two edges incident to s cross. Call this augmented graph G'_φ. Note that, due to the crossings, edges may be subdivided into several pieces, which we call *arcs*.

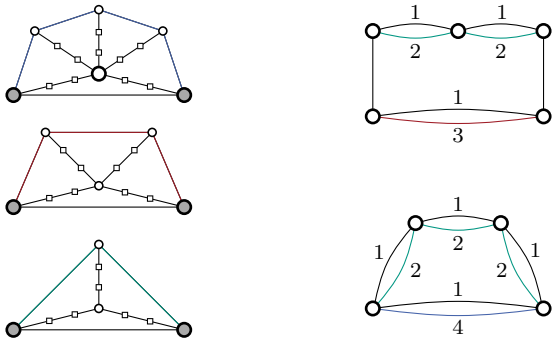
In the following we will construct gadgets modeling a flow-like problem on G'_φ. Each variable has $2k - 1$ units of flow, where d is the degree in G'_φ. It sends one unit of flow into each incident edge. For the remaining units of flow, it takes a decision. Either it sends the remaining flow to the sink (value `false`), or it evenly distributes it to all incident edges leading to a clause (value `true`). We then construct gadgets

for the arcs, which simply pass on the information from one end to the other and crossing gadgets, which pass the information over crossings. Here it is crucial that the crossover happens between information flows of different sizes. The only crossings happen between variable–clause connections, which carry either one or two units of flow and variable–sink connections, which carry either one or three/four units of flow (depending on the degree of the variable). Since our construction is such that flows cannot be split, this allows to cross over these information flows. The clauses gadgets are constructed such that there is a face that has size d if and only if it receives as incoming flow the number of incident variables plus one, which models the fact that precisely one of them must be assigned the truth value `true`.

Next, we observe that for a satisfying truth assignment, there are precisely C satisfied literals and $L-C$ unsatisfied literals in the formula. Each satisfied literal ensures that only one unit is sent towards the sink, whereas each unsatisfied literal ensures that two units are sent towards the sink. Thus, there are precisely $C+2(L-C) = 2L-C$ units of flow sent to s via n edges. We design a gadget that admits an embedding where every face has size d no matter how the incoming flow is distributed to the edges incident to s, we call this the *sink gadget*. Let now H_φ denote the graph obtained from G_φ by replacing each variable, arc, crossing, clause, and the sink by a corresponding gadget. To ensure that the embedding of H_φ follows the embedding of G_φ, we triangulate each face of H_φ that corresponds to a face of G_φ and then insert into each triangle a construction that ensures that each of the internal faces has size d. This fixes the planar embedding of H_φ except for the decisions that are modeled by the gadgets. It is then clear that H_φ admits a planar embedding if an only if φ is satisfiable.

We now give a more detailed overview of the construction. The basic tool for passing information are wheels whose outer cycle has d vertices for $d = 3, 4, 5$, and whose inner edges are subdivided $(k-1)/2$ times such that all inner faces have size k; see Fig. 5.2(a). Note that this is possible since k is odd. We then designate two adjacent vertices of the outer cycle as *poles* u and v, where it attaches to the rest of the graph. The flip of this gadget then decides with of the two face incident to its outside is incident to a path of length 1 and which is incident to a path of length $d-1$. We call these two paths the *boundary paths*. In this respect, and since there internal faces always have size k, and hence are not relevant, these constructions behave like a single edge where one side has length 1 and the other one has length $d-1$. We therefore call them $(1,2)$-, $(1,3)$- *and* $(1,4)$-*edges*, respectively. We use them to model the flows from the above description.

We are now ready to describe our gadgets. A variable of degree d in G'_φ (recall that $d = 2$ or $d = 3$), the gadget is a cycle of length $k-d$ such that d edges are $(1,2)$-edges (the *output edges*) and one is an $(1, d-1)$-edge (the *sink edge*); see Fig. 5.2(b)



(a) (b)

Figure 5.2: Illustration of the gadgets for the proof of Theorem 5.9 in the case $d = 7$. (a) $(1, 2)$-edge, $(1, 3)$-edge, and $(1, 4)$-edge; the poles are shaded, the subdivision vertices are small squares. (b) Variable gadgets for variables occurring two (above) and three times (below), respectively. The $(1, k)$-edges are represented thick and the numbers give the lengths of the respective boundary paths. Note that simultaneously exchanging the numbers at each edge also gives an embedding where the inner face has size d, and these are the only two such choices. The edge at the bottom is the sink edge, the $(1, 2)$-edges are the output edges.

for variable gadgets for $k = 7$. Clearly, for the inner face f to have size k, either all $(1, 2)$-edges must be embedded such that their boundary paths of length 2 are incident to it and the $(1, d-1)$-edge must be embedded such that its boundary path of length 1 is incident to f, or all $(1, 2)$-edges and the $(1, k-1)$-edge must be flipped. This precisely models the flows emanated by a variable as described above.

We use *pipe gadgets* to transport flow along an arc. By construction, each arc transports flows from exactly one of the three sets $\{1, 2\}$, $\{1, 3\}$ and $\{1, 4\}$. We give separate pipes for them. Let the set of flow values be $\{1, d\}$. The gadget is a cycle of length $k - d + 1$ where two nonadjacent edges are $(1, d)$-edges, one *input* and one *output* edge. Clearly, there are $k-d-1$ edges contributing length 1 to the inner face of the gadgets. Thus, the two $(1, d)$-edges must together contribute paths of length $d+1$, which occurs if and only if the information encoded by the input edge is transferred to the output edge.

For a clause of degree d in G_φ (recall that $d = 2$ or $d = 3$), the gadget is a cycle of length $k - 1$ where d edges are $(1, 2)$-edges. Obviously, the inner face f has size k if

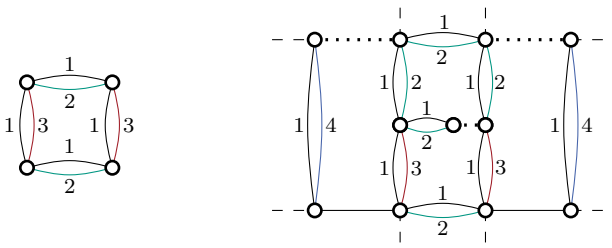


Figure 5.3: Illustration of the crossing gadgets for the proof of Theorem 5.9 in the case $d = 7$. (a) A crossing gadget for two edges, one carrying values in $\{1, 2\}$ and one in $\{1, 3\}$. Observe that simultaneously exchanging the numbers at opposite edges results again in an inner face of size d, and this can be done independently for both pairs. (b) A crossing gadget for two edges, one carrying values in $\{1, 2\}$ and the other in $\{1, 4\}$. The dashed lines show where further gadgets attach, the length of the dotted paths depends on d, for $d = 7$ their length would be 0.

and only if precisely one of the $(1, 2)$-edges has its boundary path of length 2 incident to f. Thus, the gadget correctly models a 1-in-3-SAT clause.

For a crossing, one of the two edges transports values in $\{1, 2\}$ and the other values in $\{1, 3\}$ or in $\{1, 4\}$. If it is $\{1, 3\}$, we simply use a cycle of length 4, where two opposite edges are $(1, 2)$-edges and the other two opposite edges are $(1, 3)$-edges; see Fig. 5.3(a). From each pair of opposite edges, we designate one as the input edge and one as the output edge. It is not hard to see that the inner face has size k if and only if the state from each input edge is correctly transferred to the output edge. Of course, the same approach could be used for the case $\{1, 4\}$, however, this would require $k \geq 8$. Instead, we use a different approach; see Fig. 5.3(b) for an illustration. First, we split the information into two separate pieces, one that transmits a value in $\{1, 2\}$ and one that transmits a value in $\{1, 3\}$ (note that the sum of the differences between the upper and the lower values remains constant). Then we cross over the part of the sink edge carrying the flow in $\{1, 3\}$ as before. To cross the part of the sink edge carrying flow in $\{1, 2\}$ with the variable–clause arc, we use a gadget we call *flow switch*. It consists of a cycle of length $k - 2$ where four edges are $(1, 2)$-edges, and for each pair of opposite $(1, 2)$-edges one is declared the input and one is the output. Note that, unlike the above crossing gadget, this does not necessarily transfer the input information to the correct output edge. It only requires that half of

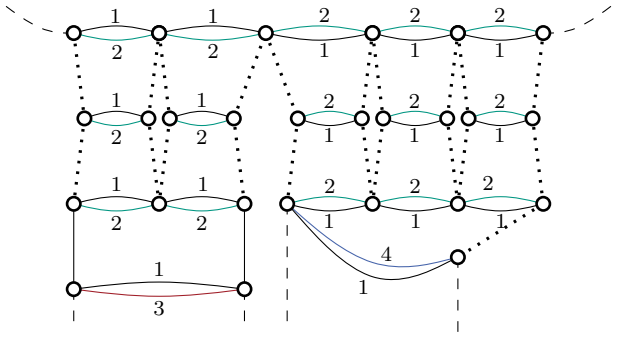


Figure 5.4: Splitting the sink edges into $(1,2)$-edges that form a cycle. The face above the construction is the inner face of the sink. The length of the dotted paths can be adjusted to accommodate for all values of $d \geq 7$ (length 0 for the one adjacent to the $(1,4)$-edge for $d = 7$).

the $(1,2)$-edges have a boundary path of length 2 in the inner face. However, the fact that, afterwards, we use a symmetric construction as for splitting the flow in $\{1,4\}$ to merge the two flows on the sink edges after the crossing back into a flow in $\{1,4\}$ enforces this behavior.

We can now construct a graph H'_φ by replacing each variable by a variable gadget, each clause by a clause gadget, each crossing by a crossing gadget and each arc by a corresponding pipe gadget. The gadgets are joined to each other by identifying corresponding input and output edges of the gadgets (i.e., we identify the corresponding construction), taking into account the embedding of G'_φ. For the sink, we first attach to each of the output edges of the pipe gadgets leading there, a corresponding variable gadget via its sink edge to split the flow arriving there into $(1,2)$-edges. We identify the endpoints of these $(1,2)$-edges such that they form a simple cycle whose interior faces represents the sink. See Fig 5.4.

We now arbitrarily triangulate, possibly by inserting vertices, all faces corresponding to a face of G'_φ that are not internal faces of a gadget and insert into each of the resulting triangles a vertex connected to each triangle vertex by a path of length $(k-1)/2$. This ensures that all resulting subfaces of the triangles have size k and at the same time the embedding of H'_φ is fixed except for the flips of the $(1,d)$-edges. Using the above arguments, it is not hard to see that φ admits a satisfying 1-in-3 truth assignment if and only if H_φ admits a planar embedding where each face has size k except for the face representing the sink vertex, which is bounded by L $(1,2)$-edges and has size $2L - C$. To complete the proof, we present a construction

for the interior of the sink that always allows an embedding where all inner faces have size k as long as the outer face has size $2L - C$, i.e., C edges have a boundary path of length 1 at the inner face, and the remaining $L - C$ have a boundary path of length 2 at the inner face.

This works in two steps. First, we build a *shift ring*, which allows to shift the information encoded in a subset of the edges by one unit to the left or the right. The shift ring consists of a ring of pairs of flow switches as shown in Fig. 5.5. Its inner

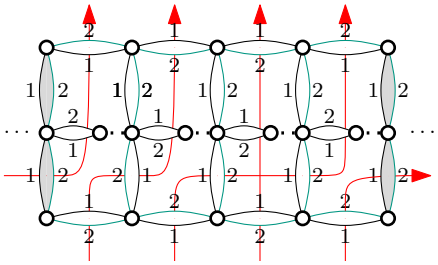


Figure 5.5: Illustration of a shift ring consisting of several flow switches. The gray edges on the left and right boundary are identified. The red arrows illustrate the flow of information, where the states of two $(1, 2)$-edges carrying different information is transposed in the circular ordering compared to the inner face of the shift ring (above the construction) and the outer face of the shift ring(below the construction).

and outer face is bounded by L $(1, 2)$-edges. There is a natural bijection between the $(1, 2)$-edges on the inner and on the outer ring, but the shift ring allows to exchange the state of two adjacent edges. We then nest sufficiently many shift rings (L^2 certainly suffice), which allows us to assume that, in the innermost face, the edges whose boundary paths have length 2 are consecutive, and the first one (in clockwise direction) is at a specific position. Second, assuming that the innermost face has the configuration of its $(1, 2)$-edges as described above, we simply triangulate it arbitrarily and insert into each triangle the construction that makes every face have size k. This concludes the construction of the sink gadget, and thus the proof. □

Theorem 5.10 *k-UNIFORMFACES is NP-complete for all even $k \geq 10$.*

Proof: The proof runs along the lines of the proof of Theorem 5.9. For this proof, however, we need the additional assumption that number L of literals is even. If it is not the case, we take a variable x that occurs only twice (subdivide an edge to introduce such a variable as in the proof of Theorem 5.8 if none exists). We then create

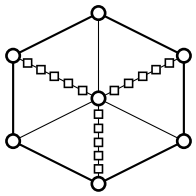


Figure 5.6: Construction for subdividing a face of even length (bold) into faces of arbitrary size d (here $d = 8$).

new variables u, v, w and add the clause (x, u, v) and twice the clause (u, v, w). If x has the value *true*, then setting $u = v = \texttt{false}$ and $w = \texttt{true}$ satisfies the new clauses, and if $x = \texttt{false}$, then $u = \texttt{true}$, $v = w = \texttt{false}$ satisfies them. Thus the resulting formula is equivalent to the original one, has an even number of literals, and satisfies all the conditions of Theorem 5.8.

Now the proof essentially reuses the construction from Theorem 5.9 for such a formula. However, since all faces have to have even size, it is not possible to construct a $(1, 2)$-edge (or $(1, k)$-edges with k even for that matter); its outer face would have to have odd length while all interior faces have even length, which is not possible. We thus use $(1, 3)$-edges to transmit information for all the gadgets. The sink edges can then use $(1, 5)$- and $(1, 7)$-edges. A crossing gadget for a $(1, 3)$-edge and a $(1, 5)$-edge requires a face of size 10. A $(1, 7)$-edge can be split into a $(1, 3)$ and a $(1, 5)$-edge for the corresponding crossing gadget. By choosing suitably long pipes, we can ensure that all faces that are not internal to a gadget have even length. Such a face can then be subdivided into faces of size d by adding a new vertex incident to all vertices of the face and subdividing every second of these edges $d - 3$ times; see Fig 5.6. For the sink, we first distribute the information to $(1, 3)$-edges using variable gadgets and then use corresponding shift rings made of flow switches for $(1, 3)$-edges. Now, in the innermost face of the shift ring, there are L $(1, 3)$-edges of which C have length 1 in the inner face and $L - C$ have length 3, and they can be assumed to be en bloc, starting at a specific edge. The total length of the innermost face then is $3(L - C) + C = 3L - 2C$, which is even due to our assumption on L. Thus, the construction making every face have size d can be done as described above. □

5.5 Open Problems.

In this chapter, we study problems MINMAXFACE and UNIFORMFACES. We show NP-hardness results for both problems as well as polynomial-time algorithms for several values of the face sizes. Observe that, our polynomial-time algorithms for MINMAXFACE when $k \leq 4$ allow us to try to answer the C-PLANARITY problem for some instances. Further, we give a 6-approximation for minimizing the maximum face in a planar embedding.

Some interesting open questions are left open. What is the complexity of k-UNIFORMFACES for $k = 5$ and $k = 8$? Are UNIFORMFACES and MINMAXFACE polynomial-time solvable for biconnected series-parallel graphs? And, more in general, are such problems FPT with respect to treewidth?

Part III

Clusters and Levels

Chapter 6

Strip Planarity Testing

In this chapter[1] we introduce and study the STRIP PLANARITY testing problem, which takes as an input a planar graph $G(V, E)$ and a function $\gamma : V \to \{1, 2, \ldots, k\}$ and asks whether a planar drawing of G exists such that each edge is represented by a curve that is monotone in the y-direction and, for any $u, v \in V$ with $\gamma(u) < \gamma(v)$, it holds that $y(u) < y(v)$.

The problem has strong relationships with some of the most deeply studied variants of the planarity testing problem, such as C-PLANARITY, UPWARD PLANARITY, and LEVEL PLANARITY.

We show that the STRIP PLANARITY testing problem is polynomial-time solvable if G has a fixed planar embedding.

6.1 Introduction

Testing the planarity of a given graph is one of the oldest and most deeply investigated problems in algorithmic graph theory. A celebrated result of Hopcroft and Tarjan [HT74] states that the planarity testing problem is solvable in linear time.

A number of interesting variants of the planarity testing problem have been considered in the literature [Sch13]. Such variants mainly focus on testing, for a given planar graph G, the existence of a planar drawing of G satisfying certain constraints. For example the *partial embedding planarity* problem [ADF+10, JKR13] asks whether a planar drawing $\mathcal{G}$ of a given planar graph G exists in which the drawing of a subgraph H of G in $\mathcal{G}$ coincides with a given drawing $\mathcal{H}$ of H. C-PLANARITY *test-*

[1] The contents of this chapter are a joint work with Patrizio Angelini, Giuseppe Di Battista, and Fabrizio Frati, appeared in [ADDF13a] and submitted to journal.

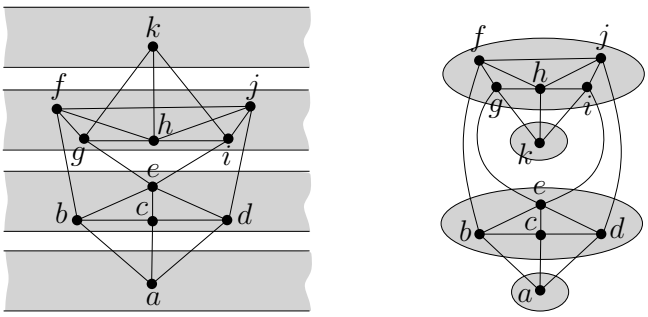


Figure 6.1: (a) A negative instance (G, γ) of the STRIP PLANARITY testing problem whose associated clustered graph $C(G, T)$ (b) is c-planar.

ing [DF09, FCE95b, JKK+09], UPWARD PLANARITY *testing* [BDLM94, GT01a, HL96], LEVEL PLANARITY *testing* [JLM98], *embedding constraints planarity testing* [GKM08], RADIAL LEVEL PLANARITY TESTING [BBF05], T-LEVEL PLANARITY *testing* [WSP12], and CL-PLANARITY *testing* [FB04] are further examples of problems falling in this category.

In this chapter we introduce and study the STRIP PLANARITY testing problem, which is defined as follows. The input of the problem consists of a planar graph $G(V, E)$ and of a function $\gamma : V \to \{1, 2, \ldots, k\}$. The problem asks whether a *strip planar* drawing of (G, γ) exists, i.e. a planar drawing of G such that each edge is represented by a curve that is monotone in the y-direction and, for any $u, v \in V$ with $\gamma(u) < \gamma(v)$, it holds $y(u) < y(v)$. The name "strip" planarity comes from the fact that, if a strip planar drawing Γ of (G, γ) exists, then k disjoint horizontal strips $\gamma_1, \gamma_2, \ldots, \gamma_k$ can be drawn in Γ so that γ_i lies below γ_{i+1}, for $1 \leq i \leq k-1$, and so that γ_i contains a vertex x of G if and only if $\gamma(x) = i$, for $1 \leq i \leq k$. It is not difficult to argue that strips $\gamma_1, \gamma_2, \ldots, \gamma_k$ can be given as part of the input, and the problem is to decide whether G can be planarly drawn so that each edge is represented by a curve that is monotone in the y-direction and each vertex x of G with $\gamma(x) = i$ lies in the strip γ_i. That is, arbitrarily predetermining the placement of the strips does not alter the possibility of constructing a strip planar drawing of (G, γ).

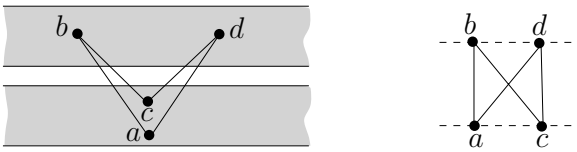


Figure 6.2: (a) A positive instance (G, γ) of the STRIP PLANARITY testing problem that is not level planar. (b) A level drawing of (G, γ) that is not level planar.

Strip Planarity and Other Planarity Variants

Before describing our results, we discuss the strong relationships of the STRIP PLANARITY testing problem with three famous graph drawing problems. Namely, we consider the C-PLANARITY, the LEVEL PLANARITY, and UPWARD PLANARITY testing problems.

STRIP PLANARITY and C-PLANARITY. An instance (G, γ) of the STRIP PLANARITY testing problem naturally defines a clustered graph $C(G, T)$, where T consists of a root having k children $\mu_1, \ldots, \mu_k$ and, for every $1 \leq j \leq k$, cluster μ_j contains every vertex x of G such that $\gamma(x) = j$. The C-PLANARITY of $C(G, T)$ is a necessary condition for the STRIP PLANARITY of (G, γ), since suitably bounding the strips in a strip planar drawing of (G, γ) provides a c-planar drawing of $C(G, T)$. However, the C-PLANARITY of $C(G, T)$ is not sufficient for the STRIP PLANARITY of (G, γ) (see Fig. 6.1). We will prove that the STRIP PLANARITY testing problem reduces in polynomial time to the C-PLANARITY testing problem. Furthermore, it turns out that STRIP PLANARITY testing *coincides* with a special case of a problem opened by Cortese et al. [CDPP05, CDPP09] and related to C-PLANARITY testing. The problem asks whether a graph G can be planarly embedded "inside" an host graph H, which can be thought as having "fat" vertices and edges, with each vertex and edge of G drawn inside a prescribed vertex and a prescribed edge of H, respectively. The STRIP PLANARITY testing problem coincides with this problem in the case in which H is a path.

STRIP PLANARITY and LEVEL PLANARITY. The LEVEL PLANARITY *testing* problem takes as an input a planar graph $G(V, E)$ and a function $\gamma : V \to \{1, 2, \ldots, k\}$ and asks whether a planar drawing of G exists such that each edge is represented by a curve that is monotone in the y-direction and each vertex $u \in V$ is drawn on the horizontal line $y = \gamma(u)$. The LEVEL PLANARITY testing (and embedding) problem is known to be solvable in linear time [JLM98], although a se-

quence of incomplete characterizations by forbidden subgraphs [FK07, HKL04] (see also [EFK09]) has revealed that the problem is not yet fully understood. The similarity of the LEVEL PLANARITY testing problem with the STRIP PLANARITY testing problem is evident: They have the same input, they both require planar drawings with y-monotone edges, and they both constrain the vertices to lie in specific regions of the plane; they only differ for the fact that such regions are horizontal lines in one case, and horizontal strips in the other one. Clearly the LEVEL PLANARITY of an instance (G, γ) is a sufficient condition for the STRIP PLANARITY of (G, γ), as a level planar drawing is also a strip planar drawing. However, it is easy to construct instances (G, γ) that are strip planar and yet not level planar, even if we require that the instances are *strict*, i.e., no edge (u, v) is such that $\gamma(u) = \gamma(v)$. See Fig. 6.2. Also, the approach of [JLM98] seems to be not applicable to test the STRIP PLANARITY of a graph. Namely, Jünger et al. [JLM98] visit the instance (G, γ) one level at a time, representing with a PQ-tree [BL76] the possible orderings of the vertices in level i that are consistent with a level planar embedding of the subgraph of G induced by levels $\{1, 2, \ldots, i\}$. However, when visiting an instance (G, γ) of the STRIP PLANARITY testing problem one strip at a time, PQ-trees seem to be not powerful enough to represent the possible orderings of the vertices in strip i that are consistent with a strip planar embedding of the subgraph of G induced by strips $\{1, 2, \ldots, i\}$.

STRIP PLANARITY and UPWARD PLANARITY. The UPWARD PLANARITY *testing* problem asks whether a given directed graph $\overrightarrow{G}$ admits an *upward planar drawing*, i.e., a drawing which is planar and such that each edge is represented by a curve monotonically increasing in the y-direction, according to its orientation. Testing the UPWARD PLANARITY of a directed graph $\overrightarrow{G}$ is an $\mathcal{NP}$-hard problem [GT01a], however it is polynomial-time solvable, e.g., if $\overrightarrow{G}$ has a fixed embedding [AHR10, BDLM94], or if it has a single-source [HL96]. A strict instance (G, γ) of the STRIP PLANARITY testing problem naturally defines a directed graph $\overrightarrow{G}$, by directing an edge (u, v) of G from u to v if $\gamma(u) < \gamma(v)$. It is easy to argue that the UPWARD PLANARITY of $\overrightarrow{G}$ is a necessary and not always sufficient condition for the STRIP PLANARITY of (G, γ) (see Fig.s 6.3(a) and 6.3(b)). Roughly speaking, in an upward planar drawing different parts of the graph are free to "nest" one into the other, while in a strip planar drawing, such a nesting is only allowed if coherent with the strip assignment.

Our Results

In this chapter, we show that the STRIP PLANARITY testing problem is quadratic-time solvable for planar graphs with a fixed plane embedding. Our approach consists

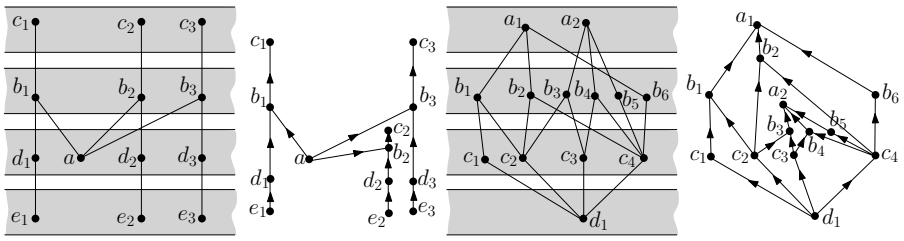


Figure 6.3: Two negative instances (G_1, γ_1) (a) and (G_2, γ_2) (b) of the STRIP PLANARITY testing problem whose associated directed graphs are upward planar, where G_1 is a tree and G_2 is a subdivision of a triconnected plane graph.

of performing a sequence of modifications to the input instance (G, γ) (such modifications consist mainly of insertions of graphs inside the faces of G) that ensure that the instance satisfies progressively stronger constraints while not altering its STRIP PLANARITY. Eventually, the STRIP PLANARITY of (G, γ) becomes equivalent to the UPWARD PLANARITY of its associated directed graph, which can be tested in quadratic time.

We also show a polynomial-time reduction from the STRIP PLANARITY testing problem (for graphs without a fixed plane embedding) to the C-PLANARITY testing problem.

The rest of the chapter is organized as follows. In Section 6.2 we present some preliminaries; in Section 6.3 we show a quadratic-time algorithm to test the STRIP PLANARITY of graphs with fixed plane embedding; in Section 6.4 we show a polynomial-time reduction from the STRIP PLANARITY testing problem to the C-PLANARITY testing problem; finally, in Section 6.5 we conclude and present open problems.

6.2 Preliminaries

In this section we present some definitions and terminology.

In this chapter we will show how to test in quadratic time whether a graph with a prescribed *plane embedding* is strip planar, where a plane embedding of a graph G is a planar embedding (or combinatorial embedding) of G together with a choice for its outer face. Since an n-vertex graph with a fixed combinatorial embedding has $O(n)$ choices for its outer face, this implies that the STRIP PLANARITY of a graph with a prescribed combinatorial embedding can be tested in cubic time. In the remainder of

this section and in Section 6.3, we will assume all the considered graphs to have a prescribed plane embedding, even when not explicitly mentioned.

We now define some concepts related to STRIP PLANARITY.

Definition 6.1 *An instance (G, γ) of* STRIP PLANARITY *is* strict *if it contains no intra-strip edge, where an edge (u, v) is* intra-strip *if $\gamma(u) = \gamma(v)$.*

Definition 6.2 *An instance (G, γ) of* STRIP PLANARITY *is* proper *if, for every edge (u, v) of G, it holds $\gamma(v) - 1 \leq \gamma(u) \leq \gamma(v) + 1$.*

For any face f of G, we denote by $C_f = (u_0, u_1, \ldots, u_l)$ the walk delimiting the boundary of f. Recall that G is not necessarily 2-connected, hence f might not be delimited by a simple cycle; also, if a vertex incident to f is a cut-vertex, it might appear several times in C_f. Consider any vertex occurrence u_j with $0 \leq j \leq l$. See Fig. 6.4(a). We say that u_j is a *local minimum* for f if $\gamma(u_j) \leq \gamma(u_{j-1})$ and $\gamma(u_j) \leq \gamma(u_{j+1})$, where indices are modulo $l + 1$. Analogously, we say that u_j is a *local maximum* for f if $\gamma(u_j) \geq \gamma(u_{j-1})$ and $\gamma(u_j) \geq \gamma(u_{j+1})$, where indices are modulo $l + 1$. Observe that several occurrences of the same vertex might be local minima or maxima for f. In the reminder of the chapter, we often say "the number of minima and maxima" of an instance (G, γ) of STRIP PLANARITY, as a short form for "the number of distinct pairs (v_j, g) such that vertex occurrence v_j is a local minimum or maximum for face g of G". Further, we say that u_j is a *global minimum* for f (a *global maximum* for f) if $\gamma(u_j) \leq \gamma(u_i)$ (resp. $\gamma(u_j) \geq \gamma(u_i)$), for every $i \neq j$ with $0 \leq i \leq l$.

Let (G, γ) be a 2-connected strict proper instance of the STRIP PLANARITY testing problem. A path $(u_1, \ldots, u_j)$ in G is *monotone* if $\gamma(u_i) = \gamma(u_{i-1}) + 1$, for every $2 \leq i \leq j$. Consider any face f; since G is 2-connected, C_f is a simple cycle. A global minimum u_m and a global maximum u_M for f are *consecutive* in f if no global minimum and no global maximum exists in one of the two paths connecting u_m and u_M in C_f. A local minimum u_m and a local maximum u_M for a face f are *visible* if one of the paths P connecting u_m and u_M in C_f is such that, for every vertex u of P, it holds $\gamma(u_m) < \gamma(u) < \gamma(u_M)$.

We conclude the section with the following definitions.

Definition 6.3 *An instance (G, γ) of* STRIP PLANARITY *is* quasi-jagged *if it is 2-connected, strict, proper and if, for every face f of G and for any two visible local minimum u_m and local maximum u_M for f, one of the two paths connecting u_m and u_M in C_f is monotone (see Fig. 6.4(b)).*

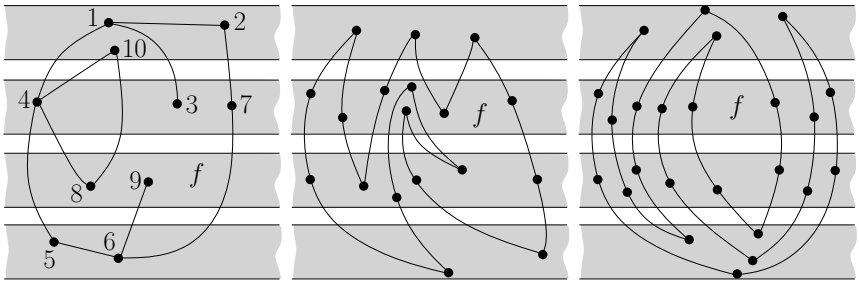


Figure 6.4: (a) The boundary of a face f in an instance of STRIP PLANARITY. The walk delimiting f is $C_f = (1, 2, 7, 6, 9, 6, 5, 4, 8, 10, 4, 1, 3, 1)$. Both occurrences of vertex 6 are local and global minima for f; vertex 8 is a local minimum and not a global minimum for f; vertex 7 is neither a local minimum nor a local maximum for f. (b) The boundary of a face f in a quasi-jagged instance of STRIP PLANARITY. (c) The boundary of a face f in a jagged instance of STRIP PLANARITY.

Definition 6.4 *An instance (G, γ) of* STRIP PLANARITY *is* jagged *if it is 2-connected, strict, proper and if, for every face f of G, any local minimum for f is a global minimum for f, and every local maximum for f is a global maximum for f (see Fig. 6.4(c)).*

Observe that a jagged instance (G, γ) is also quasi-jagged.

6.3 How To Test Strip Planarity

In this section we show an algorithm to test STRIP PLANARITY. In Sections 6.3–6.3, we will assume every considered STRIP PLANARITY instance to be connected. We will show in Section 6.3 how to extend our polynomial-time algorithm to non-connected instances.

In Section 6.3 we show how to reduce a general instance to an equivalent set of strict instances. In Section 6.3 we show how to reduce a strict instance to an equivalent strict proper instance. In Section 6.3 we show how to reduce a strict proper instance to an equivalent 2-connected strict proper instance. In Section 6.3 we show how to reduce a 2-connected strict proper instance to an equivalent quasi-jagged instance. In Section 6.3 we show how to reduce a quasi-jagged instance to an equivalent jagged instance. Finally, in Section 6.3 we show that testing the STRIP PLANARITY

of a jagged instance is equivalent to test the UPWARD PLANARITY of the associated directed graph.

From a General Instance to a Strict Instance

In this section we show how to reduce a general instance of the STRIP PLANARITY testing problem to an equivalent set of strict instances.

Lemma 6.1 *Let (G, γ) be an instance of the* STRIP PLANARITY *testing problem with n vertices, k strips, and r minima and maxima.*

There exists an $O(n^2)$-time algorithm that either decides that (G, γ) is not strip planar, or constructs a set $S = \{(G_1^*, \gamma_1^*), \ldots, (G_m^*, \gamma_m^*)\}$ of strict instances such that:

- *(G, γ) is strip planar if and only if all of $(G_1^*, \gamma_1^*), \ldots, (G_m^*, \gamma_m^*)$ are strip planar;*
- *the total number of vertices of instances $(G_1^*, \gamma_1^*), \ldots, (G_m^*, \gamma_m^*)$ is linear in n, i.e., $\sum_{i=1}^{m} |G_i^*| \in O(n)$;*
- *(G_i^*, γ_i^*) has at most $k + 1$ strips, for each $1 \leq i \leq m$; and*
- *the total number of minima and maxima over all instances (G_i^*, γ_i^*) is linear in r.*

In order to prove Lemma 6.1, we show an algorithm that receives a non-strict instance (G, γ) of STRIP PLANARITY with n vertices, k strips, and r minima and maxima, and either decides that (G, γ) is not strip planar or applies one of *Operations 1–3*, to be described below, to construct one or two "simpler" STRIP PLANARITY instances out of (G, γ).

First, since (G, γ) has k strips, we can assume that there exist vertices x and y with $\gamma(x) = 1$ and $\gamma(y) = k$, as if strips γ_1 and γ_k were empty, then they could be removed from (G, γ) without changing its STRIP PLANARITY. Second, the algorithm checks in $O(n)$ time whether there exist vertices x' and y' incident to the outer face of G such that $\gamma(x') = 1$ and $\gamma(y') = k$. If the test fails, then the algorithm concludes that (G, γ) is not strip planar. We prove the correctness of this step of the algorithm. Assume that no vertex x' exists with $\gamma(x') = 1$, the case in which no vertex y' exists with $\gamma(y') = k$ being analogous. By assumption, a vertex x such that $\gamma(x) = 1$ exists; however, since all the edges of G are represented by y-monotone curves in any strip planar drawing of (G, γ), it follows that the interior of the possibly non-simple cycle $\mathcal{C}$ delimiting the outer face of G has no intersection with γ_1, hence either x is outside

$\mathcal{C}$ (not preserving the given plane embedding of G) or x is not in γ_1 (not satisfying the requirements of a strip planar drawing).

If vertices x' and y' incident to the outer face of G exist with $\gamma(x') = 1$ and $\gamma(y') = k$, then the algorithm proceeds as follows.

If there exists an intra-strip edge (u, v) that is part of a *separating triangle* (u, v, z) in G, i.e., a 3-cycle that contains vertices both in its interior and in its exterior in G, then the algorithm applies **Operation 1**, that is, it constructs two STRIP PLANARITY instances (G', γ') and (G'', γ'') defined as follows. Denote by G' the plane subgraph of G induced by the vertices lying outside cycle (u, v, z) together with u, v, and z; also, denote by G'' the plane subgraph of G induced by the vertices lying inside cycle (u, v, z) together with u, v, and z. Also, let $\gamma'(x) = \gamma(x)$, for every vertex x in G', and let $\gamma''(x) = \gamma(x)$, for every vertex x in G''. We have the following:

Claim 6.1 *The following statements hold:*

- (G, γ) *is strip planar if and only if* (G', γ') *and* (G'', γ'') *are both strip planar;*
- $|G'| + |G''| = n + 3$*;*
- *the number of strips of each of* (G', γ') *and* (G'', γ'') *is not larger than* k*; and*
- *the total number of minima and maxima of* (G', γ') *and* (G'', γ'') *is at most* $r + 6$.

Proof: The second and third statement are trivial. The fourth statement is proved by observing that G' and G'' have the same faces of G, except for the two faces, one in G' and one in G'', delimited by cycle (u, v, z).

We prove the first statement. The necessity is trivial, given that G' and G'' are subgraphs of G, that $\gamma(x) = \gamma'(x)$, for every vertex x of G', and that $\gamma(x) = \gamma''(x)$, for every vertex x of G''. The sufficiency is proved as follows. Suppose that (G', γ') and (G'', γ'') admit strip planar drawings Γ' and Γ'', respectively. Scale Γ'' so that it fits inside the drawing of cycle (u, v, z) in Γ'. Suitably stretch the edges of G'' in Γ'' so that: (i) the drawing of cycle (u, v, z) in Γ'' coincides with the drawing of cycle (u, v, z) in Γ', (ii) no two edges in Γ'' cross, and (iii) each vertex x of G'' lies in the strip associated with $\gamma''(x)$. Then, the drawing Γ obtained by gluing Γ' and Γ'' along cycle (u, v, z) is a strip planar drawing of (G, γ). □

If there exists an intra-strip edge (u, v) in (G, γ) that is not part of a separating triangle in G, and if the outer face of G is not delimited by 3-cycle (u, v, z), for some vertex z, then the algorithm applies **Operation 2**, that is, it constructs a STRIP PLANARITY instance (G', γ') defined as follows. Graph G' is constructed from G

by *contracting* edge (u, v). That is, identify u and v to be the same vertex w, whose incident edges are all the edges incident to u and v, except for (u, v); the clockwise order of the edges incident to w is: All the edges incident to u in G in the same clockwise order starting at (u, v), and then all the edges incident to v in G in the same clockwise order starting at (v, u). If u and v share any common neighbor z in G (in this case cycle (u, v, z) delimits an internal face of G, given that (u, v, z) is not a separating triangle and that the outer face of G is not delimited by (u, v, z)), then just one edge (w, z) is introduced in G'. Since G is plane, G' is plane; by construction, G' is simple. Let $\gamma'(x) = \gamma(x)$, for every vertex $x \neq u, v$ in G, and let $\gamma'(w) = \gamma(u)$. We have the following.

Claim 6.2 *The following statements hold:*

- (G, γ) *is strip planar if and only if* (G', γ') *is strip planar;*
- $|G'| = n - 1$*;*
- *the number of strips of* (G', γ') *is* k*; and*
- *the number of minima and maxima of* (G', γ') *is at most* r*.*

Proof: The second and third statement are trivial. The fourth statement is proved by observing that if w is a local minimum or maximum for a face g, then at least one of u and v is a local minimum or maximum for g. We next prove the first statement.

We first prove the necessity. Consider any strip planar drawing Γ of (G, γ) (see Fig. 6.5(a)). Assume that $2 \leq \gamma(u) \leq k - 1$. Denote by $p_1, p_2, \ldots, p_h$ and by $q_1, q_2, \ldots, q_l$ the left-to-right order of the intersection points of the edges of G with the lines delimiting strip $\gamma(u)$ from the top and from the bottom, respectively. Insert dummy vertices at points $p_1, p_2, \ldots, p_h$ and $q_1, q_2, \ldots, q_l$. Each of such vertices splits an edge of G into two dummy edges, one inside $\gamma(u)$ and one outside it. Insert dummy edges (p_1, q_1), (p_h, q_l), (p_i, p_{i+1}), for $1 \leq i \leq h-1$, and (q_i, q_{i+1}), for $1 \leq i \leq l-1$, in $\gamma(u)$. Contract edge (u, v) into a single vertex w. Triangulate the internal faces of the resulting plane graph H by inserting dummy vertices and edges, so that no edge connects two vertices p_i and p_j or q_i and q_j with $j \geq i + 2$ (see Fig. 6.5(b)). Construct a convex straight-line drawing of H in which vertices $p_1, p_2, \ldots, p_h$ and $q_1, q_2, \ldots, q_l$ have the same positions they have in Γ (see Fig. 6.5(c)). Such a drawing always exists [CYN84]. Slightly perturb the positions of the vertices different from $p_1, p_2, \ldots, p_h$ and $q_1, q_2, \ldots, q_l$, so that no two vertices have the same y-coordinate. As a consequence, the edges of H different from (p_i, p_{i+1}), for $1 \leq i \leq h - 1$, and (q_i, q_{i+1}), for $1 \leq i \leq l - 1$, are y-monotone curves. Removing the inserted dummy vertices and edges results in a strip planar drawing of (G', γ') (see Fig. 6.5(d)).

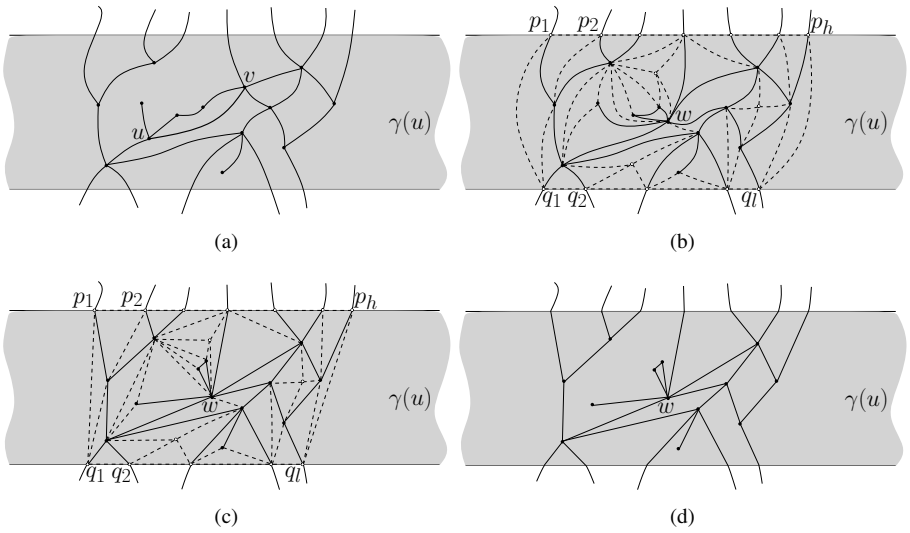


Figure 6.5: (a) A strip planar drawing Γ of (G, γ). (b) Modifications performed on the part of G inside $\gamma(u)$, resulting in an internally-triangulated simple plane graph H. (c) A convex straight-line drawing of H. (d) A strip planar drawing of (G', γ').

The cases in which $\gamma(u) = 1$ or $\gamma(u) = k$ can be handled analogously to the case in which $2 \leq \gamma(u) \leq k - 1$. Namely, assume that $\gamma(u) = 1$. Insert dummy vertices at points $p_1, p_2, \ldots, p_h$ as before (points $q_1, q_2, \ldots, q_l$ are now not defined). Also insert points p_0 and p_{h+1} to the left of p_1 and to the right of p_h, respectively. Moreover, insert a dummy vertex d in γ_1 and insert dummy edges (p_0, d), (p_{h+1}, d), and (p_i, p_{i+1}), for $0 \leq i \leq h$. Contract edge (u, v) into a single vertex w. Triangulate the internal faces of the plane graph H whose outer face is delimited by cycle $(d, p_0, p_1, \ldots, p_{h+1})$ by inserting dummy vertices and edges, so that no edge connects two vertices p_i and p_j with $j \geq i + 2$. Construct a convex straight-line drawing of H in which vertices $p_0, p_1, \ldots, p_{h+1}$ have the same positions they have in Γ and in which d is at any point inside γ_1. Such a drawing always exists [CYN84]. Slightly perturb the positions of the vertices different from $p_0, p_1, \ldots, p_{h+1}$ so that no two of them have the same y-coordinate. As a consequence, the edges of H different from (p_i, p_{i+1}), for $0 \leq i \leq h$, are y-monotone curves. Removing the inserted dummy vertices and edges results in a strip planar drawing of (G', γ'). Finally, the case in which $\gamma(u) = k$ is symmetric to the case in which $\gamma(u) = 1$.

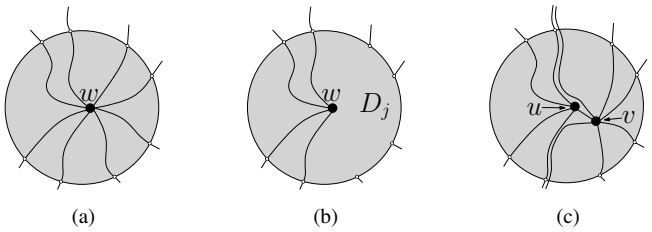


Figure 6.6: (a) A disk D containing w. (b) Region D_j. (c) Drawing edge (u, v) and the edges incident to v inside D.

We now prove the sufficiency. Consider any strip planar drawing Γ' of (G', γ'). Slightly perturb the positions of the vertices in Γ', so that no two vertices have the same y-coordinate. Consider a disk D containing w, small enough so that it contains no vertex different from w, and it contains no part of an edge that is not incident to w (see Fig. 6.6(a)). Remove from the interior of D the parts of the edges incident to w that correspond to edges incident to v. The edges still incident to w partition D into regions $D_1, D_2, \ldots, D_l$. At most one of such regions, say D_j, has to contain edges incident to w corresponding to edges incident to v (see Fig. 6.6(b)). In fact, all the edges incident to w corresponding to edges incident to v appear consecutively around w in G'. Insert a y-monotone curve incident to w in D_j. Let v be the end-vertex of such a curve different from w. Rename w to u. Draw y-monotone curves connecting v with the intersection points of the boundary of D_j with the edges incident to w that used to lie inside D_j. Also, if u and v share one or two neighbors in G, say that u and v share two neighbors z and z' in G, the other case being analogous, then cycles (u, v, z) and (u, v, z') delimit internal faces of G. The two edges (v, z) and (v, z') can be drawn arbitrarily close to the edges (w, z) and (w, z'), which are the edges that delimit D_j (see Fig. 6.6(c)). The resulting drawing Γ is a strip planar drawing of (G, γ). □

If Operations 1 and 2 do not apply, then G contains no separating triangle, the outer face of G is delimited by a 3-cycle (u, v, z), and only edges (u, v), (u, z), and (v, z) can possibly be intra-strip edges. By assumption, (G, γ) contains at least one intra-strip edge, hence at least one of (u, v), (u, z), and (v, z), say (u, v), is an intra-strip edge. If $\gamma(u) = \gamma(v) = \gamma(z)$, then since the algorithm positively checked whether there exist vertices x' and y' incident to the outer face of G such that $\gamma(x') = 1$ and $\gamma(y') = k$, we have that $k = 1$, hence every vertex of (G, γ) belongs to γ_1, and (G, γ) is strip planar. Otherwise, we have that (u, v) is the only intra-strip edge of

(G, γ). Then, we have that either $\gamma(u) = \gamma(v) = 1$ and $\gamma(z) = k > 1$, or that $\gamma(u) = \gamma(v) = k > 1$ and $\gamma(z) = 1$; assume the former, as the other case can be discussed analogously.

In this case the algorithm applies **Operation 3**, that is, it constructs a STRIP PLANARITY instance (G', γ') defined as follows. Graph G' coincides with graph G. Further, $\gamma'(t) = \gamma(t) + 1$, for every vertex $t \neq u$ in G', and $\gamma'(u) = \gamma(u)$. That is, (G', γ') coincides with (G, γ), except that vertex u is moved to a strip "preceding" γ_1. We have the following.

Claim 6.3 *The following statements hold:*

- (G, γ) *is strip planar if and only if* (G', γ') *is strip planar;*
- (G', γ') *is a strict instance of* STRIP PLANARITY*;*
- $|G'| = n$*;*
- *the number of strips of* (G', γ') *is* $k + 1$*; and*
- *the number of minima and maxima of* (G', γ') *is less than* r*.*

Proof: The third and fourth statement are trivial. The fifth statement follows from the fact that v is a local minimum for the two faces f^1_{uv} and f^2_{uv} incident to edge (u, v) in (G, γ), while it is not a local minimum for f^1_{uv} and f^2_{uv} in (G', γ'). The second statement is true because (u, v) is the only intra-strip edge of (G, γ). We prove the first statement.

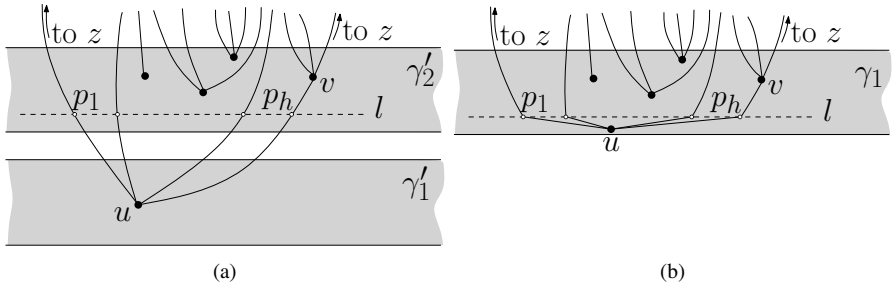


Figure 6.7: (a) Strip planar drawing Γ' of (G', γ'). (b) Strip planar drawing Γ of (G, γ).

We first prove the sufficiency. Consider any strip planar drawing Γ' of (G', γ') (see Fig. 6.7(a)). Let l be any horizontal line in γ'_2 below every vertex in γ'_2. Since u is the only vertex in γ'_1, it follows that u is the only vertex below l. Denote by $p_1, \ldots, p_h$ the intersection points of the edges incident to u with l. Move u from its position in Γ' to any point in γ'_2 below l; further, redraw the line segments connecting u with $p_1, \ldots, p_h$ as straight-line segments. The resulting drawing Γ is a strip planar drawing of (G, γ) (see Fig. 6.7(b)).

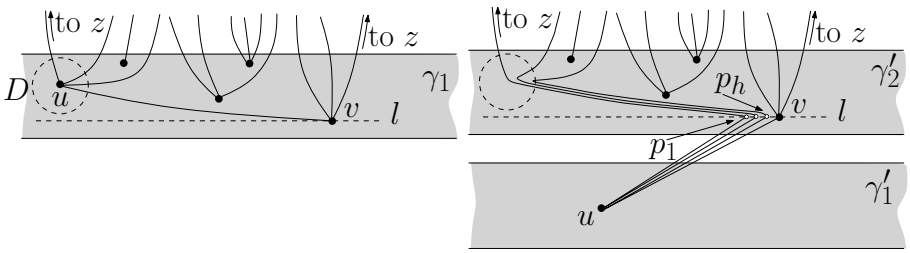


Figure 6.8: (a) Strip planar drawing Γ of (G, γ). (b) Strip planar drawing Γ' of (G', γ').

We now prove the necessity. Consider any strip planar drawing Γ of (G, γ). Since cycle (u, v, z) delimits the outer face of G, and since every edge of G is represented by a y-monotone curve in Γ, we have that either u or v is the vertex of G with smallest y-coordinate in Γ. In the former case, consider any horizontal line l above u and below every other vertex of G. Denote by $p_1, \ldots, p_h$ the intersection points of the edges incident to u with l. Move u from its position in Γ to any point in γ'_1; further, redraw the line segments connecting u with $p_1, \ldots, p_h$ as straight-line segments. The resulting drawing Γ' is a strip planar drawing of (G', γ'). In the latter case (see Fig. 6.8(a)), consider the horizontal line l through v and consider a small disk D centered at u, small enough so that it contains no vertex different from u, and it contains no part of an edge that is not incident to u. Delete from Γ edge (u, v) and the parts of the other edges incident to u lying inside D. Redraw the latter curves as y-monotone curves between their intersection point with the boundary of D and a point on l; such curves are arbitrarily close to the drawing of edge (u, v) in Γ. Denote by $p_1, \ldots, p_h, v$ the intersection points of the edges incident to u with l. Move u from its position in Γ to any point in γ'_1; further, redraw the line segments connecting u with $p_1, \ldots, p_h, v$ as straight-line segments. The resulting drawing Γ' is a strip planar drawing of (G', γ')

(see Fig. 6.8(b)). □

Claims 6.1–6.3 allow us to prove Lemma 6.1, as shown in the following.

First, with an $O(n)$-time preprocessing, we determine whether each edge of G is intra-strip or not. We construct and maintain a set S of STRIP PLANARITY instances, initialized as $S = \{(G, \gamma)\}$. When every graph in S is strict, we return S.

As long as there exists a non-strict instance (G_i, γ_i) in S, we check in $O(n)$ time whether there exist vertices x' and y' incident to the outer face of G_i such that $\gamma_i(x') \leq \gamma_i(x)$, for every x in G_i, and such that $\gamma_i(y') \geq \gamma_i(y)$, for every y in G_i. If the test fails we conclude that (G_i, γ_i) and hence (G, γ) is not strip planar; otherwise, we apply one of Operations 1–3.

Consider (if it exists) any internal intra-strip edge (u, v) or any external intra-strip edge (u, v), provided that the outer face of G_i is not delimited by a cycle (u, v, z). We check whether (u, v) belongs to a separating triangle; this check can be accomplished in $O(n)$ time as follows: For each vertex z in G_i different from u and v, we check in $O(1)$ time whether edges (u, z) and (v, z) exist or not, and in case they do whether cycle (u, v, z) delimits a face of G_i or not.

If (u, v) does not belong to any separating triangle, then Operation 2 applies to (G_i, γ_i). Hence, an equivalent instance (G_i', γ_i') is obtained in $O(n)$ time containing one less vertex than (G_i, γ_i), the same number of strips as (G_i, γ_i), and at least one less intra-strip edge than (G_i, γ_i). Replace (G_i, γ_i) with (G_i', γ_i') in S.

If (u, v) is an edge of a separating triangle (u, v, z), then Operation 1 applies to (G_i, γ_i). Graphs (G_i', γ_i') and (G_i'', γ_i'') can be constructed in $O(n)$ time by performing two traversals of G_i. Replace (G_i, γ_i) with (G_i', γ_i') and (G_i'', γ_i'') in S.

If Operations 1–2 do not apply, then the outer face of G_i is delimited by a 3-cycle (u, v, z), such that (u, v) is an intra-strip edge. If $\gamma(u) = \gamma(v) = \gamma(z)$, then (G_i, γ_i) is strip planar if and only if G_i is planar, which is always the case by the assumptions on the input; then, we remove (G_i, γ_i) from S. If $\gamma(u) = \gamma(v) \neq \gamma(z)$, then Operation 3 applies to (G_i, γ_i). Hence, a strict instance (G_i', γ_i') is obtained in $O(n)$ time containing the same number of vertices as (G_i, γ_i). Replace (G_i, γ_i) with (G_i', γ_i') in S.

Note that $O(n)$ applications of Operations 1–3 are required in order to obtain a set S that contains only strict instances. This, together with Claims 6.1–6.3, implies that set S eventually satisfies the properties required by the statement of Lemma 6.1. Since Operations 1–3 can be performed in $O(n)$ time, the total running time of the algorithm is $O(n^2)$. This concludes the proof of Lemma 6.1.

From a Strict Instance to a Strict Proper Instance

In this section we show how to reduce a strict instance of the STRIP PLANARITY testing problem to an equivalent strict proper instance.

Lemma 6.2 *Let (G, γ) be a strict instance of the* STRIP PLANARITY *testing problem with n vertices, k strips, and r minima and maxima.*

There exists an $O(kn)$-time algorithm that constructs an equivalent strict proper instance (G^*, γ^*) with $O(kn)$ vertices, k strips, and r minima and maxima.

Proof: We define graph G^* as follows. Initialize G^* as the empty graph whose vertex set is the same as the one of G; also, let $\gamma^*(v) = \gamma(v)$ for every vertex v of G^* and G. Then, consider any edge (u, v) of G such that $\gamma(u) = \gamma(v)+j$, for some $j \geq 1$. Insert a path $(v = u_1, u_2, \ldots, u_{j+1} = u)$ in G^* such that $\gamma^*(u_{i+1}) = \gamma^*(u_i) + 1$, for every $1 \leq i \leq j$. The repetition of this operation for every edge (u, v) of G results in a proper strict instance (G^*, γ^*) of the STRIP PLANARITY testing problem. Since each edge of G corresponds to a path with at most k vertices, G^* has $O(kn)$ vertices. Also, the strips of (G^*, γ^*) coincide with the strips of (G, γ); hence, (G^*, γ^*) has k strips. Further, a distinct pair (v, g) such that v is a local minimum or maximum for face g of G^* exists if and only if the same pair exists in G; hence, (G^*, γ^*) has r such pairs. Moreover, the construction can clearly be performed in $O(kn)$ time.

Finally, (G^*, γ^*) is strip planar if and only if (G, γ) is. Namely from a strip planar drawing Γ^* of (G^*, γ^*) a strip planar drawing Γ of (G, γ) can be obtained by representing each edge (u, v) of G with the same curve representing the path $(v = u_1, u_2, \ldots, u_{j+1} = u)$ in Γ^*. Conversely, from a strip planar drawing Γ of (G, γ) a strip planar drawing Γ^* of (G^*, γ^*) can be obtained by representing path $(v = u_1, u_2, \ldots, u_{j+1} = u)$ with the same curve representing edge (u, v) in Γ where, for each $2 \leq i \leq j$, vertex u_i is placed at any point in the intersection of strip $\gamma^*(u_i)$ and the curve representing edge (u, v) in Γ. ar □

From a Strict Proper Instance to a 2-Connected Strict Proper Instance

In this section we show how to reduce a strict proper instance of the STRIP PLANARITY testing problem to an equivalent 2-connected strict proper instance.

Lemma 6.3 *Let (G, γ) be a strict proper instance of the* STRIP PLANARITY *testing problem with n vertices, k strips, and r minima and maxima.*

There exists an $O(n)$-time algorithm that constructs an equivalent 2-connected strict proper instance (G^*, γ^*) with $O(n)$ vertices, k strips, and $O(r)$ minima and maxima.

Proof: Let $(G(V, E), \gamma)$ be a strict proper instance of the STRIP PLANARITY testing problem. First, we associate each edge $e \in E$ with the unique block of G it belongs to and with its two incident faces. This computation can be performed in total $O(n)$ time [Tar72]. Denote by b the number of blocks of G. Note that, if $b = 1$, then (G, γ) is a 2-connected strict proper instance. Otherwise, consider any cutvertex c of G.

Let $e_1, e_2, \ldots, e_m, e_{m+1} = e_1$ be the clockwise order of the edges incident to c; let $e_i = (c, v_i)$ and $e_{i+1} = (c, v_{i+1})$ be two edges belonging to distinct blocks B_p and B_q of G, respectively, for some $1 \leq i \leq m$. Let f be the face of G that is to the left of e_i when traversing this edge from v_i to c. Insert a vertex w and edges (w, v_i) and (w, v_{i+1}) inside f. Let $V' = V \cup \{w\}$ and let $E' = E \cup \{(w, v_i), (w, v_{i+1})\}$; also, let G' be the graph (V', E'). Let $\gamma' : V' \to \{1, 2, \ldots, k\}$ be defined as follows: $\gamma'(u) = \gamma(u)$ for every vertex $u \in V$, and $\gamma'(w) = \gamma(c)$.

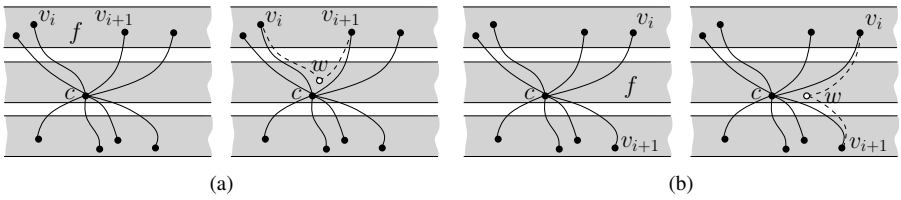


Figure 6.9: Inserting vertex w and edges (w, v_i) and (w, v_{i+1}) inside f if (a) $\gamma(v_i) = \gamma(v_{i+1}) = \gamma(c) + 1$ and if (b) $\gamma(v_i) = \gamma(c) + 1 = \gamma(v_{i+1}) + 2$.

We claim that (G', γ') is an instance of the STRIP PLANARITY testing problem that is equivalent to (G, γ). We first prove that the claim implies the lemma, and we then prove the claim. Refer to Fig. 6.9.

First, (G', γ') is proper and strict, given that (G, γ) is proper and strict, that $\gamma'(w) = \gamma'(c) = \gamma'(v_i) \pm 1$, and that $\gamma'(w) = \gamma'(c) = \gamma'(v_{i+1}) \pm 1$; further, the number of blocks of G' is equal to $b - 1$, since blocks B_p and B_q of G belong to the same block of G'. Hence, the repetition of the above augmentation eventually leads to a 2-connected strict proper instance (G^*, γ^*) that is equivalent to (G, γ) and that has $|G^*| = b - 1 + n \in O(n)$ vertices. The fact that (G^*, γ^*) contains $O(r)$ minima and maxima descends from the fact that G^* has the same faces of G, except for the two faces obtained by splitting face f with path (v_i, w, v_{i+1}), and that w is incident to exactly two faces of G^*. Finally, the augmentation of (G, γ) to (G', γ') can be easily performed in $O(1)$ time (observe that, after the augmentation is performed, blocks B_p

and B_q are given the same name, that is now associated to every edge in each of these blocks, in $O(1)$ time). Hence, the total running time is $O(n)$ given that $b \in O(n)$.

We now prove the claim. One direction is trivial. Namely, if (G', γ') is strip planar, then (G, γ) is strip planar, given that G is a subgraph of G' and given that $\gamma(u) = \gamma'(u)$ for every $u \in V$. We prove the other direction. Assume that (G, γ) is strip planar and let Γ be any strip planar drawing of (G, γ). Draw edges (v_i, w) and (v_{i+1}, w) in Γ as y-monotone curves lying inside f and arbitrarily close to edges (v_i, c) and (v_{i+1}, c), respectively, in such a way that w lies in a point arbitrarily close to c in $\gamma'(c)$. The resulting drawing Γ' of G' is strip planar. Namely, each vertex u of G' lies inside $\gamma'(u)$ (by assumption if $u \in V$ and by construction if $u = w$); further, each edge e of G' is represented by a y-monotone curve (by assumption if $e \in E$ and by construction if e is incident to w); finally, Γ' is planar because Γ is planar (by assumption) and because edges (v_i, w) and (v_{i+1}, w) do not cross any edge of G, given that they lie inside f and that they are arbitrarily close to the drawing of edges (v_i, c) and (v_{i+1}, c), respectively, which do not cross any edge of G (by assumption).

This concludes the proof of the claim and hence of the lemma. □

From a 2-Connected Strict Proper Instance to a Quasi-Jagged Instance

In this section we show how to reduce a 2-connected strict proper instance of the STRIP PLANARITY testing problem to an equivalent quasi-jagged instance.

Lemma 6.4 *Let (G, γ) be a 2-connected strict proper instance of the* STRIP PLANARITY *testing problem with n vertices, k strips, and r minima and maxima.*

There exists an $O(kr + n)$-time algorithm that constructs an equivalent quasi-jagged instance (G^*, γ^*) with $O(kr + n)$ vertices, k strips, and $O(r)$ minima and maxima.

Consider any face f of G containing two visible local minimum and maximum u_m and u_M, respectively, such that no path connecting u_m and u_M in C_f is monotone. Insert a monotone path connecting u_m and u_M inside f. Denote by (G^+, γ^+) the resulting instance of the STRIP PLANARITY testing problem. We have the following claim:

Claim 6.4 *(G^+, γ^+) is strip planar if and only if (G, γ) is strip planar.*

Proof: One direction of the equivalence is trivial, namely if (G^+, γ^+) is strip planar, then (G, γ) is strip planar, since G is a subgraph of G^+ and $\gamma(v) = \gamma^+(v)$ for every vertex v in G.

We prove the other direction. Consider a strip planar drawing Γ of (G, γ). Slightly perturb the positions of the vertices in Γ so that no two of them have the same y-coordinate. Denote by P and Q the two paths connecting u_m and u_M along C_f. Since u_m and u_M are visible, it holds $\gamma(u_m) < \gamma(v) < \gamma(u_M)$ for every internal vertex v of P or for every internal vertex v of Q. Assume that $\gamma(u_m) < \gamma(v) < \gamma(u_M)$ holds for every internal vertex v of P, the other case being analogous. We also assume w.l.o.g. that face f is to the right of P when traversing such a path from u_m to u_M. We modify Γ, if necessary, while maintaining its STRIP PLANARITY so that a y-monotone curve $\mathcal{C}$ connecting u_m and u_M can be drawn inside f.

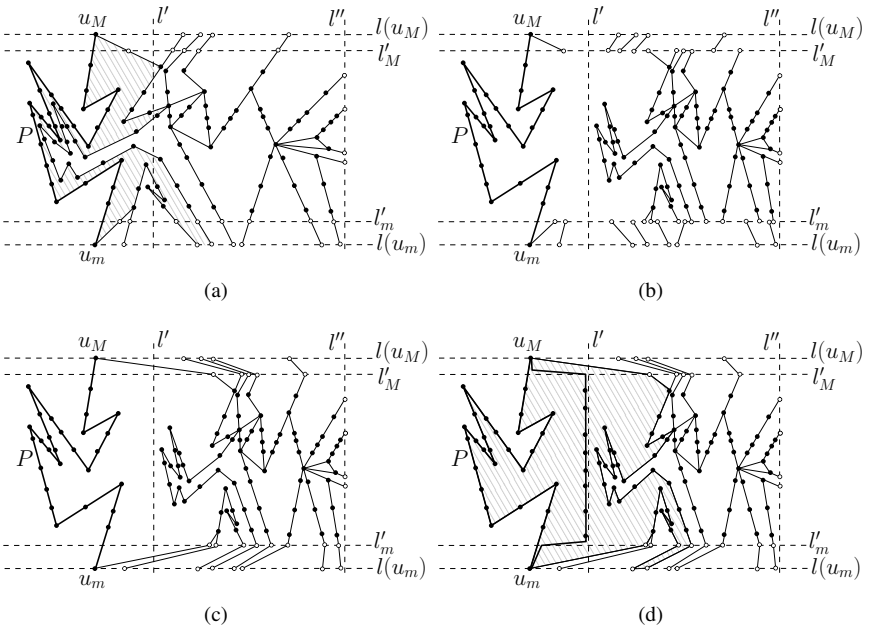


Figure 6.10: (a) Drawing Γ inside region R. The part of face f inside R is colored gray. Path P is represented by a thick line. Intersection points of edges with lines l'', $l(u_m)$, $l(u_M)$, l'_m, and l'_M are represented by white circles. (b) Drawing Γ inside region R after the shrinkage. (c) Reconnecting parts of edges that have been disconnected by the shrinkage. (d) Drawing of a monotone path connecting u_m and u_M inside f.

We introduce some notation. Refer to Fig. 6.10(a). Let $l(u_m)$ and $l(u_M)$ be the horizontal lines through u_m and u_M, respectively. Let l' and l'' be vertical lines entirely lying to the right of P, with l'' to the right of l'. Denote by D the distance between l' and l''. Denote by R the bounded region of the plane delimited by P, by $l(u_m)$, by $l(u_M)$, and by l''. Denote by l'_m (by l'_M) an horizontal line above $l(u_m)$ (resp. below $l(u_M)$) and sufficiently close to $l(u_m)$ (resp. to $l(u_M)$) so that the strip delimited by l'_m and $l(u_m)$ (resp. by l'_M and $l(u_M)$) does not contain any vertex of G other than u_m (resp. other than u_M). Finally, we define some regions inside R. Let R' be the bounded region of the plane delimited by P, by l'_m, by l'_M, and by l'; let R'' be the bounded region of the plane delimited by P, by l'_m, by l'_M, and by l''; let R''' be the bounded region of the plane delimited by l', by l'_m, by l'_M, and by l''; let R_B be the bounded region of the plane delimited by P, by l'_m, by $l(u_m)$, and by l''; and let R_A be the bounded region of the plane delimited by P, by l'_M, by $l(u_M)$, and by l''. We are going to modify Γ in such a way that no vertex and no part of an edge lies in the interior of R'. The part of Γ outside R is not modified in the process.

We perform a horizontal shrinkage of the part of Γ that lies in the interior of R'' (the vertices of P stay still). This is done in such a way that every intersection point of an edge with l'' keeps the same position, and the distance between l'' and every point in the part of Γ that used to lie inside R'' becomes strictly smaller than D. See Fig. 6.10(b). Hence, the part of Γ that used to lie inside R'' is now entirely contained in R'''. However, some edges of G (namely those that used to intersect l'_m and l'_M) are now disconnected; e.g., if an edge of G used to intersect l'_m, now such an edge contains a line segment inside R''', which has been shrunk, and a line segment inside R_B, whose drawing has not been modified by the shrinkage. By construction R_B does not contain any vertex in its interior. Hence, the line segments that lie in R_B form in Γ a planar y-monotone matching between a set A of points on l'_m and a set B of points on $l(u_m)$. As a consequence of the shrinkage, the position of the points in A has been modified, however their relative order on l'_m has not been modified. Thus, we can delete the line segments in R_B and reconnect the points in B with the new positions of the points in A on l'_m so that each edge is y-monotone and no two edges intersect. See Fig. 6.10(c). After performing an analogous modification in R_A, we obtain a planar y-monotone drawing Γ' of G in which no vertex and no part of an edge lies in the interior of R'. Since no vertex changed its y-coordinate and every edge is y-monotone, Γ' is a strip planar drawing of (G, γ).

Finally, we draw a y-monotone curve $\mathcal{C}$ connecting u_m and u_M. This is done as follows. See Fig. 6.10(d). Starting from u_m, follow path P, slightly to the right of it, until reaching line l'_m; then, continue drawing $\mathcal{C}$ with a line segment increasing in the x-direction and slightly increasing in the y-direction, until reaching a point arbitrarily close to l'; then, continue drawing $\mathcal{C}$ with a vertical line segment that stops

at l'_M; then, continue drawing $\mathcal{C}$ with a line segment decreasing in the x-direction and slightly increasing in the y-direction, until reaching a point arbitrarily close to path P; finally, follow path P until reaching u_M. Place each vertex x of the monotone path connecting u_m and u_M on $\mathcal{C}$ at a suitable y-coordinate, so that x lies in the strip $\gamma(x)$. We thus obtained a strip planar drawing of (G^+, γ^+), which concludes the proof. □

Claim 6.4 implies Lemma 6.4, as proved in the following.

First, the repetition of the above described augmentation leads to a quasi-jagged instance (G^*, γ^*). In fact, whenever the augmentation is performed, the resulting instance is clearly strict, proper, and 2-connected; further, the number of triples (v_m, v_M, g) such that vertices v_m and v_M are visible local minimum and maximum for face g, respectively, and such that both paths connecting v_m and v_M along C_f are not monotone decreases at least by 1, thus eventually the number of such triples is zero, and the instance is quasi-jagged.

Second, we prove that (G^*, γ^*) can be constructed from (G, γ) in $O(kr+n)$ time, that $|G^*| \in O(kr+n)$, and that there are $O(r)$ minima and maxima in (G^*, γ^*). These statements easily descend from the following two arguments.

- The insertion of a monotone path connecting a local minimum v_m with a local maximum v_M for a face g can be easily performed in $O(\gamma(v_M) - \gamma(v_m)) = O(k)$ time, as it consists of introducing $\gamma(v_M) - \gamma(v_m) - 1$ new vertices and $\gamma(v_M) - \gamma(v_m)$ new edges in the graph. Further, whenever the insertion is performed, the number of vertices of the graph increases by $\gamma(v_M) - \gamma(v_m) - 1 \in O(k)$, and the number of distinct pairs (v, g) such that v is a local minimum or maximum for a face g of the graph increases by $O(1)$, given that only vertices v_m and v_M and only the two faces incident to the inserted path can generate new such pairs.

- The number of times the described augmentation is performed is $O(r)$. To prove this claim, it suffices to prove that the number of paths that are inserted in a face g of G is linear in the number of local minima and maxima for g. No two paths P_1 and P_2 are inserted in g connecting a vertex a_m with a vertex a_M and connecting a vertex b_m with a vertex b_M, respectively, such that a_m, b_m, a_M, and b_M appear in this circular order along the boundary of g, as when the second path insertion is performed, the two end-vertices of the path would not be incident to the same face in g. It follows that the graph that has one vertex for each local minimum or maximum for g and one edge between two vertices if a path between them has been inserted in g is planar (in fact outerplanar), hence it has a number of edges that is linear in the number of maxima and minima for g. Thus, the claim follows.

Third, (G^*, γ^*) is an instance of the STRIP PLANARITY testing problem that is equivalent to (G, γ). This directly comes from repeated applications of Claim 6.4.

This concludes the proof of Lemma 6.4.

From a Quasi-Jagged Instance to a Jagged Instance

In this section we show how to reduce a quasi-jagged instance of the STRIP PLANARITY testing problem to an equivalent jagged instance.

Lemma 6.5 *Let (G, γ) be a quasi-jagged instance of the* STRIP PLANARITY *testing problem with n vertices, k strips, and r minima and maxima.*

There exists an $O(kr + n)$-time algorithm that constructs an equivalent jagged instance (G^*, γ^*) with $O(kr + n)$ vertices, k strips, and $O(r)$ minima and maxima.

Consider any face f of G that contains some local minimum or maximum which is not a global minimum or maximum for f, respectively. Assume that f contains a local minimum v which is not a global minimum for f. The case in which f contains a local maximum which is not a global maximum for f can be discussed analogously. Denote by u (denote by z) the first global minimum or maximum for f that is encountered when walking along C_f starting at v while keeping f to the left (resp. to the right).

We distinguish two cases, namely the case in which u is a global minimum for f and z is a global maximum for f (*Case 1*), and the case in which u and z are both global maxima for f (*Case 2*). The case in which u is a global maximum for f and z is a global minimum for f, and the case in which u and z are both global minima for f can be discussed symmetrically.

In Case 1, denote by Q the path connecting u and z in C_f and containing v. Refer to Fig. 6.11(a). Consider the internal vertex v' of Q that is a local minimum for f and such that $\gamma(v') = \min_{u'} \gamma(u')$ among all the internal vertices u' of Q that are local minima for f. Traverse Q starting from u, until a vertex v'' is found with $\gamma(v'') = \gamma(v')$. Notice that, the subpath of Q between u and v'' is monotone. Insert a monotone path connecting v'' and z inside f. See Fig. 6.11(b). Denote by (G^+, γ^+) the resulting instance of the STRIP PLANARITY testing problem. We have the following claim:

Claim 6.5 *Suppose that Case 1 is applied to a quasi-jagged instance (G, γ) to construct an instance (G^+, γ^+). Then, (G^+, γ^+) is strip planar if and only if (G, γ) is strip planar. Also, (G^+, γ^+) is quasi-jagged.*

Proof: We prove that (G^+, γ^+) is strip planar if and only if (G, γ) is strip planar.

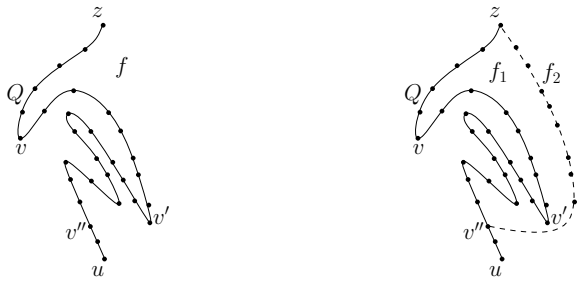


Figure 6.11: Augmentation of (G, γ) inside a face f in Case 1. (a) Before the augmentation. (b) After the augmentation.

One direction of the equivalence is trivial, namely if (G^+, γ^+) is strip planar, then (G, γ) is strip planar, since G is a subgraph of G^+ and $\gamma(x) = \gamma^+(x)$, for every vertex x in G.

We prove the other direction. Consider a strip planar drawing Γ of (G, γ). Observe that, since u and z are consecutive global minimum and maximum for f, they are visible. Since Q is not monotone, by assumption, and since (G, γ) is quasi-jagged, it follows that the path P connecting u and z in C_f and not containing v is monotone. Hence, u and z are the only global minimum and maximum for f, respectively.

For every local minimum u' in Q such that $\gamma(u') = \gamma(v')$ (including v'), define $R(u')$ to be the bounded region delimited by the two edges incident to u' in Q, and by the horizontal line delimiting $\gamma(u')$ from the top; vertically shrink $R(u')$ and the part of Γ inside it so that the y-coordinate of u' is larger than the one of v''. Observe that such a modification does not alter the STRIP PLANARITY of Γ.

Next, we distinguish two cases.

In the first case, f is an internal face of G. See Fig. 6.12(a). We draw a y-monotone curve $\mathcal{C}$ connecting v'' and z as follows. Draw a line segment of $\mathcal{C}$ inside f starting at v'' and slightly increasing in the y-direction, until reaching path P. Then, follow such a path to reach z. Place each vertex x of the monotone path connecting v'' and z on $\mathcal{C}$ at a suitable y-coordinate, so that x lies in the strip $\gamma(x)$.

In the second case, f is the outer face of G. See Fig. 6.12(b). Then, we draw a y-monotone curve $\mathcal{C}$ connecting v'' and z as follows. Draw a line segment of $\mathcal{C}$ inside f starting at v'' and slightly increasing in the y-direction, until reaching an x-coordinate which is larger than the maximum x-coordinate of any point of Γ. Then, continue drawing $\mathcal{C}$ as a vertical line segment, until a point is reached whose y-coordinate is

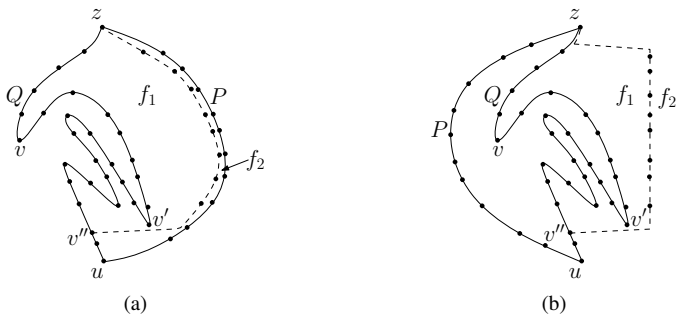


Figure 6.12: Inserting a monotone path connecting v'' and z inside f if: (a) f is an internal face, and (b) f is the outer face.

smaller than the y-coordinate of z and larger than the one of every vertex of Q different from z (recall that z is the only global maximum for f). Then, continue drawing $\mathcal{C}$ slightly increasing in the y-direction and decreasing in the x-direction, until the edge of Q incident to z is reached. Then, follow such an edge to reach z. Place each vertex x of the monotone path connecting v'' and z on $\mathcal{C}$ at a suitable y-coordinate, so that x lies in the strip $\gamma(x)$.

It remains to prove that (G^+, γ^+) is quasi-jagged. Clearly, (G^+, γ^+) is strict, proper, and 2-connected. Every face $g \neq f$ of G has not been altered by the augmentation inside f, hence, for any two visible local minimum u_m and local maximum u_M for g, one of the two paths connecting u_m and u_M in g is monotone. Denote by f_1 and f_2 the two faces into which f is split by the insertion of the monotone path connecting v'' and z, where f_1 is the face delimited by such a monotone path and by the subpath of Q between v'' and z. Face f_2 is delimited by two monotone paths, hence the only pair of visible local minimum and local maximum for f_2 is connected by a monotone path in C_{f_2}. Face f_1, on the other hand, contains a local minimum that is not a local minimum for f, namely v''. However, the existence of a local maximum u'' for f such that v'' and u'' are visible and are not connected by a monotone path in C_{f_1} would imply that u and u'' are a pair of visible local minimum and local maximum for f that is not connected by a monotone path in C_f, which contradicts the fact that (G, γ) is quasi-jagged. □

In Case 2, denote by M a *maximal* path that is part of C_f, whose end-vertices are two global maxima u_M and v_M for f, that contains v in its interior, and that does not contain any global minimum in its interior (see Fig. 6.13(a)). By the assumptions of

Case 2, such a path exists. Assume, w.l.o.g., that face f is to the right of M when walking along M starting at u_M towards v_M. Possibly $u_M = u$ and/or $v_M = z$. Let u_m (v_m) be the global minimum for f such that u_m and u_M (resp. v_m and v_M) are consecutive global minimum and maximum for f. Possibly, $u_m = v_m$. Denote by P the path connecting u_m and u_M along C_f and not containing v. Also, denote by Q the path connecting v_m and v_M along C_f and not containing v. Since M contains a local minimum among its internal vertices, and since (G, γ) is quasi-jagged, it follows that P and Q are monotone.

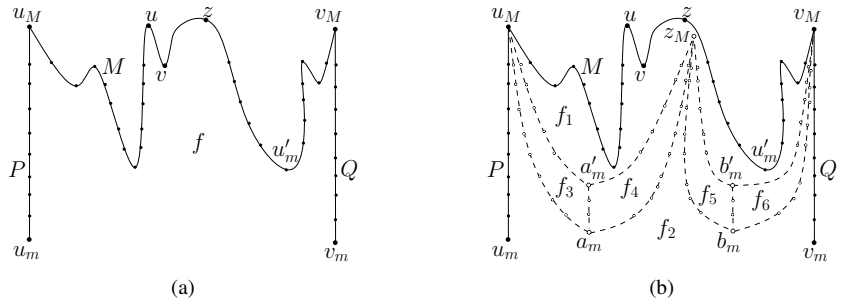


Figure 6.13: Augmentation of (G, γ) inside a face f in Case 2. (a) Before the augmentation. (b) After the augmentation.

Insert the plane graph $A(u_M, v_M, f)$ depicted by white circles and dashed lines in Fig. 6.13(b) inside f. Consider a local minimum $u'_m \in M$ for f such that $\gamma(u'_m) = \min_{v'_m} \gamma(v'_m)$ among the local minima v'_m for f in M. Set $\gamma(z_M) = \gamma(u_M)$, set $\gamma(a_m) = \gamma(b_m) = \gamma(u_m)$, and set $\gamma(a'_m) = \gamma(b'_m) = \gamma(u'_m)$. The dashed lines connecting a_m and u_M, connecting a'_m and u_M, connecting a_m and z_M, connecting a'_m and z_M, connecting b_m and z_M, connecting b'_m and z_M, connecting b_m and v_M, connecting b'_m and v_M, connecting a_m and a'_m, and connecting b_m and b'_m represent monotone paths. Denote by (G^+, γ^+) the resulting instance of the STRIP PLANARITY testing problem. We have the following claim:

Claim 6.6 *Suppose that Case 2 is applied to a quasi-jagged instance (G, γ) to construct an instance (G^+, γ^+). Then, (G^+, γ^+) is strip planar if and only if (G, γ) is strip planar. Also, (G^+, γ^+) is quasi-jagged.*

Proof: One direction of the equivalence is trivial, namely if (G^+, γ^+) is strip planar, then (G, γ) is strip planar, since G is a subgraph of G^+ and $\gamma(v) = \gamma^+(v)$ for every vertex v in G.

We prove the other direction. Consider a strip planar drawing Γ of (G, γ). Slightly perturb the position of the vertices in Γ so that no two of them have the same y-coordinate. We assume w.l.o.g. that face f is to the right of P when traversing such a path from u_m to u_M. Denote by l_M the line delimiting strip $\gamma(u_M)$ from below; also, denote by l_m the line delimiting strip $\gamma(u_m)$ from above. Further, denote by l'_m a line above l_m and sufficiently close to l_m so that the horizontal strip delimited by these two lines does not contain any vertex of G.

The proof distinguishes two cases. In the first case (Case 2A), the intersection of P with l_M lies to the left of the intersection of Q with l_M. In the second case (Case 2B), the intersection of P with l_M lies to the right of the intersection of Q with l_M. Since P and Q are represented in Γ by y-monotone curves that do not intersect each other, in Case 2A the intersection of P with l_m lies to the left of the intersection of Q with l_m, while in Case 2B the intersection of P with l_m lies to the right of the intersection of Q with l_m. In both cases, we modify Γ, if necessary, while maintaining its STRIP PLANARITY so that plane graph $A(u_M, v_M, f)$ can be planarly drawn in f with y-monotone edges.

We first discuss Case 2A.

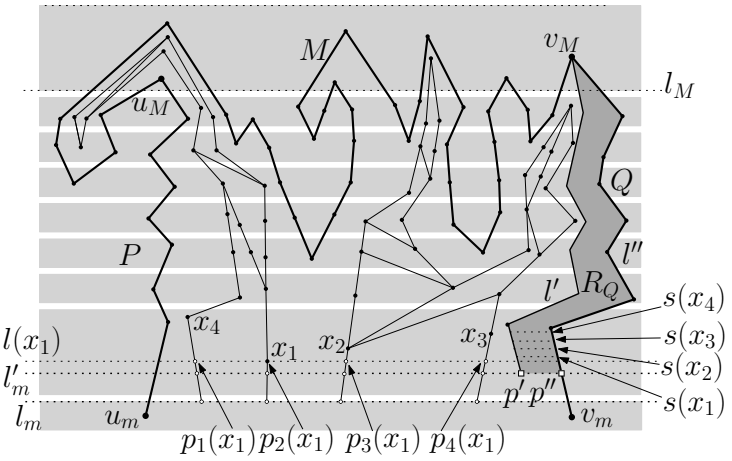


Figure 6.14: Illustration for the proof of Claim 6.6. Paths P, M, and Q are represented by thick lines. The part of the graph that is outside region R is not shown.

We introduce some notation. Refer to Fig. 6.14. Denote by R the bounded region of the plane delimited by P, by M, by Q, and by l_m. Drawing Γ will be only modified in the interior of R. Denote by R' the bounded region of the plane delimited by P,

by M, by Q, and by l'_m. We define a closed bounded region R_Q of the plane inside R as follows. Region R_Q is delimited by two y-monotone curves l' and l'' from the left and from the right, respectively, where l'' is the part of Q delimited by v_M and by the intersection point p'' of Q with l'_m, and where l' connects v_M with a point p' on l'_m, slightly to the left of l''; curves l' and l'' share no point other than v_M; region R_Q contains no vertex and no part of an edge of G in its interior, that is, the interior of R_Q entirely belongs to f. Observe that a region R_Q with such properties always exists. The part of Γ that lies in the interior of R' will be redrawn so that it entirely lies in R_Q.

For each vertex x of G that lies in the interior of R, consider the horizontal line $l(x)$ through x. Let $p_1(x), p_2(x), \ldots, p_{f(x)}(x)$ be the left-to-right order of the intersection points of edges of G with $l(x)$, where x is also a point $p_i(x)$ for some $1 \leq i \leq f(x)$. We draw a horizontal segment $s(x)$ inside R_Q, in such a way that: (i) $s(x)$ is contained in the strip $\gamma(x)$, (ii) $s(x)$ connects a point in l' with a point in l'', and (iii) if vertices x_1 and x_2 inside R are such that $y(x_1) < y(x_2)$, then $s(x_1)$ lies below $s(x_2)$. For each vertex x of G that lies in the interior of R, insert points $p'_1(x), p'_2(x), \ldots, p'_{f(x)}(x)$ in this left-to right order on $s(x)$.

Also, let $p_1(l'_m), p_2(l'_m), \ldots, p_{f(l'_m)}(l'_m)$ be the left-to-right order of the intersection points of edges of G with l'_m. Insert points $p'_1(l'_m), p'_2(l'_m), \ldots, p'_{f(l'_m)}(l'_m)$ in this left-to right order on segment $\overline{p'p''}$.

We now redraw in R_Q the vertices and edges that are inside R in Γ. Refer to Fig. 6.15.

For any line segment that is part of an edge of G and that connects two points $p_i(x_1)$ and $p_j(x_2)$, with $x_1 \neq x_2$, (or a point $p_i(l'_m)$ with a point $p_j(x)$) draw a line segment connecting $p'_i(x_1)$ and $p'_j(x_2)$ (resp. connecting $p'_i(l'_m)$ with $p'_j(x)$) inside R_Q. Observe that, if such a line segment exists, then $s(x_1)$ and $s(x_2)$ (resp. $\overline{pp'}$ and $s(x)$) are consecutive horizontal segments in R_Q. Further, the line segments connecting points on two consecutive line segments $s(x_1)$ and $s(x_2)$ (resp. $\overline{pp'}$ and $s(x)$) can be drawn as y-monotone curves inside R_Q so that they do not cross each other, give that the relative order of the points $p'_i(x)$ on $s(x)$ preserves the order of the points $p_i(x)$ on $l(x)$, for every vertex x of G in the interior of R, and the relative order of the points $p'_i(l'_m)$ on $\overline{pp'}$ preserves the order of the points $p_i(l'_m)$ on l'_m.

For each edge e that has non-empty intersection with R, delete from Γ the part e_R of e inside R. If e used to intersect l'_m, denote by $p_i(l_m)$ and $p_i(l'_m)$ the intersection points of e with l_m and l'_m before e_R was removed. Draw a y-monotone curve connecting point $p'_i(l'_m)$ on $\overline{pp'}$ with point $p_i(l_m)$. Such curves can be drawn without introducing crossings, given that the relative order of the points $p'_i(l'_m)$ on $\overline{pp'}$ preserves the order of the points $p_i(l'_m)$ on l'_m.

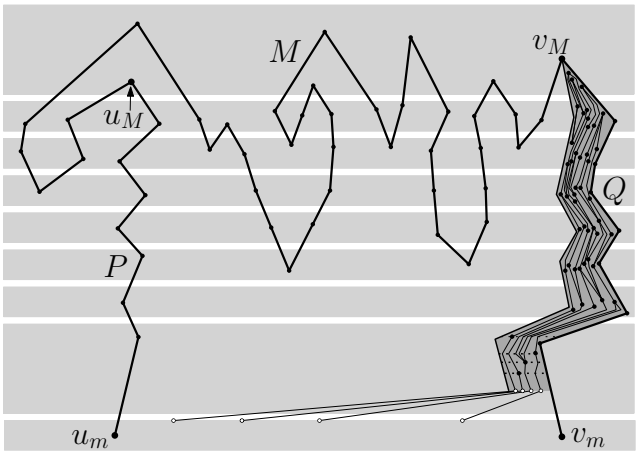


Figure 6.15: Redrawing in R_Q the vertices and edges that are inside R in Γ.

We are now ready to draw $A(u_M, v_M, f)$. Refer to Fig. 6.16. Draw the monotone path connecting v_M with b_m as a y-monotone curve $\mathcal{C}$ as follows. Place b_m in $\gamma(b_m)$ arbitrarily close to P and to l_m; follow P arbitrarily close to it until reaching l'_m; then, continue $\mathcal{C}$ with a line segment increasing in the x-direction and slightly increasing in the y-direction, until reaching l'; then complete $\mathcal{C}$ by following l' slightly to the left of it, until reaching v_M. The monotone paths connecting v_M with b'_m and connecting b_m with b'_m are arbitrarily close to the monotone path connecting v_M with b_m, slightly to the left of it; the y-coordinate of b'_m is smaller than the y-coordinate of every vertex of M. Draw the monotone path connecting b_m with z_M as a y-monotone curve arbitrarily close to P. Draw the monotone path connecting b'_m with z_M as a y-monotone curve $\mathcal{C}'$ as follows. Start drawing $\mathcal{C}'$ from b'_m with a line segment decreasing in the x-direction and slightly increasing in the y-direction, until reaching the monotone path connecting b_m and z_M; then follow such a path, slightly to the right of it, until reaching z_M. The remaining monotone paths lie arbitrarily close to P, slightly to the right of it, and arbitrarily close to the monotone path connecting b_m and z_M, slightly to the left of it.

We now discuss Case 2B.

We introduce some notation. See Fig. 6.17. Denote by l'_t the horizontal line passing through the vertex w_M of M with largest y-coordinate, and denote by l_t an horizontal line in $\gamma(u_M)$ slightly above l'_t, and close enough to l'_t so that no vertex lies in the interior of the strip delimited by l_t and l'_t. Observe that all the vertices and edges of M, of P, and of Q are entirely below l'_t, except for vertex w_M. Let $s(w_M)$ be the

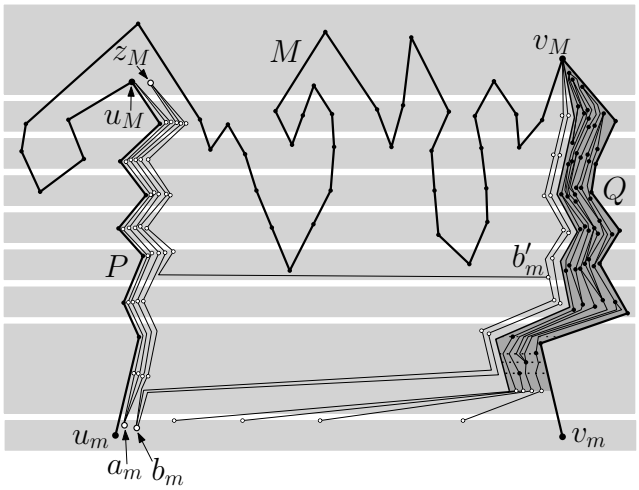


Figure 6.16: Drawing $A(u_M, v_M, f)$ (vertices are white circles and edges are solid thin lines).

vertical segment connecting w_M with l_t. Denote by l'_p and by l''_p vertical lines entirely to the right of M, P, and Q, with l''_p to the right of l'_p. Also, denote by l'_q and by l''_q vertical lines entirely to the left of M, P, and Q, with l''_q to the left of l'_q. Let R_A be the region delimited by l_t, by l'_t, by l''_p, and by l''_q. Denote by D_p and D_q the distance between l'_p and l''_p and the distance between l'_q and l''_q, respectively. Denote by R_p the bounded region of the plane delimited by l_m, by l''_p, by l_t, by P, by the part of M connecting u_M with w_M, and by $s(w_M)$. Also, denote by R_q the bounded region of the plane delimited by l_m, by l''_q, by l_t, by Q, by the part of M connecting v_M with w_M, and by $s(w_M)$. Drawing Γ will be only modified in the interior of $R_p \cup R_q$. In particular, the vertices of G and the intersection points of the edges of G with the lines delimiting $R_p \cup R_q$ will maintain the same position after the modification.

We define some regions inside R_p. Let R'_p be the bounded region of the plane delimited by l'_m, by l'_p, by l'_t, by P, and by the part of M connecting u_M with w_M; let R''_p be the bounded region of the plane delimited by l'_m, by l''_p, by l'_t, by P, and by the part of M connecting u_M with w_M; let R'''_p be the bounded region of the plane delimited by l'_m, by l''_p, by l'_p, and by l'_t; finally, let $R_{B,p}$ be the bounded region of the plane delimited by l'_m, by l''_p, by P, and by l_m.

We analogously define some regions inside R_q. Let R'_q be the bounded region of the plane delimited by l'_m, by l'_q, by l'_t, by Q, and by the part of M connecting v_M

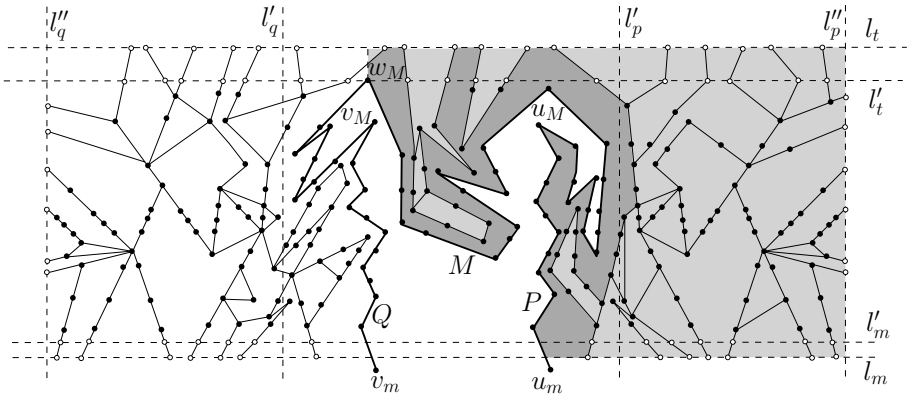


Figure 6.17: Drawing Γ inside region $R_p \cup R_q$. Region R_p is colored light and dark gray. In particular, part of face f inside R_p is colored dark gray. Paths P, Q, and M are represented by thick lines. Intersection points of edges with lines l''_p, l''_q, l_m, l'_m, l_t, and l'_t are represented by white circles.

with w_M; let R''_q be the bounded region of the plane delimited by l'_m, by l''_q, by l'_t, by Q, and by the part of M connecting v_M with w_M; let R'''_q be the bounded region of the plane delimited by l'_m, by l''_q, by l'_q, and by l'_t; finally, let $R_{B,q}$ be the bounded region of the plane delimited by l'_m, by l''_q, by Q, and by l_m.

We are going to modify Γ in such a way that no vertex and no part of an edge lies in the interior of $R'_p \cup R'_q$. The part of Γ outside $R_p \cup R_q$ is not modified in the process. This modification is similar to the one performed for the proof of Claim 6.4. Refer to Fig. 6.18.

We perform a horizontal shrinkage of the part of Γ that lies inside R''_p (the vertices and edges of P and M stay still). This is done in such a way that every intersection point of an edge with l''_p keeps the same position, and the distance between l''_p and every point in the part of Γ that used to lie inside R''_p becomes strictly smaller than D_p. Hence, the part of Γ that used to lie inside R''_p is now entirely contained in R'''_p, that is the interior of R'_p contains no vertex and no part of an edge. However, some edges of G (namely those that used to intersect l'_m and l'_t) are now disconnected; e.g., if an edge of G used to intersect l'_m, now such an edge contains a line segment inside R'''_p, which has been shrunk, and a line segment inside $R_{B,p}$, whose drawing has not been modified by the shrinkage. By construction $R_{B,p}$ does not contain any vertex in its interior. Hence, the line segments that lie in $R_{B,p}$ form in Γ a planar y-monotone matching between a set A_p of points on l'_m and a set B_p of points on l_m.

As a consequence of the shrinkage, the position of the points in A_p has been modified, however their relative order on l'_m has not been modified. Thus, we can delete the line segments in $R_{B,p}$ and reconnect the points in B_p with the new positions of the points in A_p on l'_m so that each edge is y-monotone and no two edges intersect.

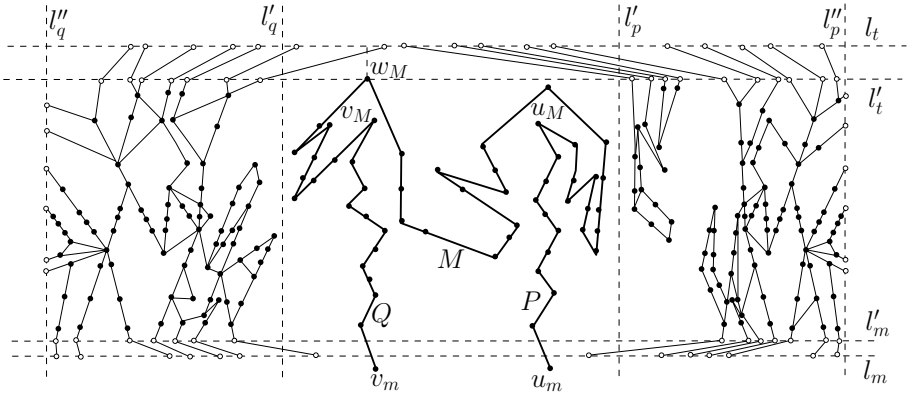


Figure 6.18: Drawing Γ' of (G, γ), obtained by shrinking the part of Γ that lies inside R''_p and inside R''_q, and by reconnecting points on l_m with points on l'_m and points on l_t with points on l'_t.

We also perform a horizontal shrinkage of the part of Γ that lies inside R''_q (the vertices and edges of Q and M stay still). This is done symmetrically to the shrinkage of the part of Γ that lies inside R''_p. As a consequence of such a shrinkage, R'_q contains no vertex and no part of an edge.

Finally, the line segments that lie in R_A form in Γ a planar y-monotone matching between a set A' of points on l'_t and a set B' of points on l_t. As a consequence of the shrinkage, the position of the points in A' has been modified, however their relative order on l'_t has not been modified. Thus, we can delete the line segments in R_A and reconnect the points in B' with the new positions of the points in A' on l'_t so that each edge is y-monotone and no two edges intersect.

We thus obtain a planar y-monotone drawing Γ' of G in which no vertex and no part of an edge lies in the interior of $R'_p \cup R'_q$. Since no vertex changed its y-coordinate and every edge is y-monotone, Γ' is a strip planar drawing of (G, γ).

We are now ready to draw $A(u_M, v_M, f)$. Refer to Fig. 6.19. Place a_m in point arbitrarily close to P, slightly to the right of it, and slightly below l_m. Draw the monotone path connecting u_M with a_m as a y-monotone curve arbitrarily close to P, and slightly to the right of it. Draw the monotone path connecting u_M with a'_m and

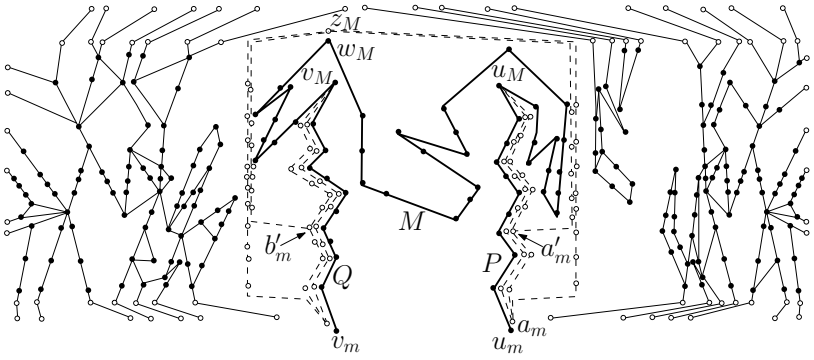


Figure 6.19: Augmentation of drawing Γ' of G with a drawing of plane graph $A(u_M, v_M, f)$.

the monotone path connecting a'_m with a_m as y-monotone curves arbitrarily close to the monotone path connecting u_M with a_m, slightly to the right of it, in such a way that a'_m has a y-coordinate smaller than the one of every vertex of P and M in $\gamma(a'_m)$. Draw the monotone path connecting a_m with z_M as a y-monotone curve $\mathcal{C}$ as follows. Starting from a_m, follow the monotone path connecting a_m with a'_m, slightly to the right of it, until reaching l'_m. Continue drawing $\mathcal{C}$ with a line segment increasing in the x-direction and slightly increasing in the y-direction. Just before reaching l'_p, stop increasing the x-coordinates along $\mathcal{C}$, and continue drawing $\mathcal{C}$ as a vertical line segment, arbitrarily close to l'_p, slightly to the left of it, until reaching l'_t. Then, finish the drawing of $\mathcal{C}$ with a line segment decreasing in the x-direction and slightly increasing in the y-direction, until reaching a point on $s(w_M)$ arbitrarily close to w_M, on which we place z_M. Draw the monotone path connecting a'_m with z_M as a y-monotone curve $\mathcal{C}'$ as follows. Starting from a'_m, draw a line segment increasing in the x-direction and slightly increasing in the y-direction, until reaching the monotone path connecting a_m with z_M. Then, follow such a path, slightly to the left of it, until reaching z_M. Finally, the drawing of the monotone paths connecting v_M with b_m, connecting v_M with b'_m, connecting b_m with b'_m, connecting b_m with z_M, and connecting b'_m with z_M are constructed symmetrically.

This concludes the construction of a strip planar drawing of (G^+, γ^+).

It remains to prove that (G^+, γ^+) is quasi-jagged. Clearly, (G^+, γ^+) is strict, proper, and 2-connected.

Every face $g \neq f$ of G has not been altered by the augmentation inside f, hence, for any two visible local minimum u_m and local maximum u_M for g, one of the two

paths connecting u_m and u_M in G is monotone. Denote by $f_1, f_2, \ldots, f_6$ the faces into which f is split by the insertion of $A(u_M, v_M, f)$ (see Fig. 6.13(b)).

Each of faces f_3, f_4, f_5, and f_6 is delimited by two monotone paths, hence, for each $i = 3, \ldots, 6$, the only pair of visible local minimum and local maximum for f_i is connected by a monotone path in C_{f_i}.

Face f_2 contains two local minima, namely a_m and b_m, and one local maximum, namely z_M, that are not incident to f in G. However, u_M and z_M are the only local maxima for f_2 that are visible with a_m; also, a_m and b_m are the only local minima for f_2 that are visible with z_M; further, z_M and v_M are the only local maxima for f_2 that are visible with b_m. For all such pairs of visible local minimum and maximum, there exists a monotone path in C_{f_2} connecting them. Finally, every pair of visible local minimum and maximum for f_2 which does not include a_m, z_M, or b_m is also a pair of visible local minimum and maximum for f, hence it is connected by the same monotone path in C_{f_2} as in C_f.

Finally, consider face f_1. Analogously as for face f_2, each of vertices a'_m, z_M, and b'_m only participates in two pairs of visible local minimum and maximum for f_1, where the second vertex of each pair is one between u_M, a'_m, z_M, b'_m, and v_M. For all such pairs, monotone paths in C_{f_1} exist by construction. Finally, every pair of visible local minimum and maximum for f_1 which does not include a'_m, z_M, or b'_m is also a pair of visible local minimum and maximum for f, hence it is connected by the same monotone path in C_{f_1} as in C_f. □

Claims 6.5–6.6 imply Lemma 6.5, as proved in the following.

First, we prove that the repetition of the above described augmentation leads to a jagged instance (G^*, γ^*) of the STRIP PLANARITY testing problem. For an instance (G, γ) and for a face g of G, denote by $n(g)$ the number of vertices that are local minima for g but not global minima for g plus the number of vertices that are local maxima for g but not global maxima for g. Also, let $n(G) = \sum_g n(g)$, where the sum is over all faces g of G. Observe that $n(G) \leq r$. We claim that, when one of the augmentations of Cases 1 and 2 is performed and instance (G, γ) is transformed into an instance (G^+, γ^+), we have $n(G^+) \leq n(G) - 1$. The claim implies that eventually $n(G^*) = 0$, hence (G^*, γ^*) is jagged.

We prove the claim. When a face f of G is augmented as in Case 1 or in Case 2, for each face $g \neq f$ and for each vertex u incident to g, vertex u is a local minimum, a local maximum, a global minimum, or a global maximum for g in (G^+, γ^+) if and only if it is a local minimum, a local maximum, a global minimum, or a global maximum for g in (G, γ), respectively. Hence, it suffices to prove that $\sum n(f_i) \leq n(f) - 1$, where the sum is over all the faces f_i that are created from the augmentation inside f.

Suppose that Case 1 is applied to insert a monotone path between vertices v'' and z inside f. Such an insertion splits f into two faces, which we denote by f_1 and f_2, as in Fig. 6.11(b). Face f_2 is delimited by two monotone paths, hence $n(f_2) = 0$. Every vertex inserted into f is neither a local maximum nor a local minimum for f_1. As a consequence, no vertex x exists such that x contributes to $n(f_1)$ and x does not contribute to $n(f)$. Further, vertex v' is a global minimum for f_1, by construction, and it is a local minimum but not a global minimum for f. Hence, v' contributes to $n(f)$ and does not contribute to $n(f_1)$. It follows that $n(f_1) + n(f_2) \leq n(f) - 1$.

Suppose that Case 2 is applied to insert plane graph $A(u_M, v_M, f)$ inside face f. Such an insertion splits f into six faces, which are denoted by $f_1, \ldots, f_6$, as in Fig. 6.13(b). Every vertex of $A(u_M, v_M, f)$ incident to a face f_i, for some $1 \leq i \leq 6$, is either a global maximum for f_i, or a global minimum for f_i, or it is neither a local maximum nor a local minimum for f_i. As a consequence, no vertex x exists such that x contributes to some $n(f_i)$ and x does not contribute to $n(f)$. Further, for each vertex x that contributes to $n(f)$, there exists at most one face f_i such that x contributes to $n(f_i)$. Finally, vertex u'_m of M is a global minimum for f_1, by construction, and it is a local minimum but not a global minimum for f. Hence, u'_m contributes to $n(f)$ and does not contribute to $n(f_i)$, for any $1 \leq i \leq 6$. It follows that $\sum_{i=1}^{6} n(f_i) \leq n(f) - 1$.

Second, the number of vertices of G^* is $O(kr + n)$. Namely, when a face f of G is augmented as in Case 1 or in Case 2, $O(k)$ vertices are inserted in G. Further, since $n(G) \leq r$, at most r augmentations are performed.

Third, the number of minima and maxima of (G^*, γ^*) is $O(r)$, since every augmentation introduces a constant number of local minima or maxima.

Fourth, (G^*, γ^*) can be constructed from (G, γ) in $O(kr + n)$ time. Namely, an $O(|g|)$-time preprocessing determines, for each face g of G, all the pairs (v, g) such that v is a local minimum for g but not a global minimum for g or v is a local maximum for g but not a global maximum for g. Hence, the preprocessing takes $O(n)$ time for the entire graph G. The augmentations of Cases 1 and 2 can be easily performed in $O(k)$ time, as they consist of introducing $O(k)$ new vertices and $O(k)$ new edges in G. Further, the number of times the described augmentation is performed is $O(r)$.

Fifth, (G^*, γ^*) is an instance of the STRIP PLANARITY testing problem that is equivalent to (G, γ). This directly comes from repeated applications of Claims 6.5 and 6.6.

This concludes the proof of Lemma 6.5.

Strip Planarity of Jagged Instances

In this section we show that testing whether a jagged instance (G, γ) of the STRIP PLANARITY testing problem is strip planar is equivalent to testing whether the associated directed graph of (G, γ) is upward planar. Based on this equivalence and on the results of the previous sections, we show that the STRIP PLANARITY testing problem for general instances can be solved in polynomial time.

Recall that the associated directed graph of (G, γ) is the directed plane graph $\overrightarrow{G}$ obtained from (G, γ) by orienting each edge (u, v) in G from u to v if and only if $\gamma(v) = \gamma(u) + 1$. We have the following:

Lemma 6.6 *A jagged instance (G, γ) of the* STRIP PLANARITY *testing problem is strip planar if and only if the associated directed graph $\overrightarrow{G}$ of (G, γ) is upward planar.*

Proof: The necessity is trivial, given that a strip planar drawing of (G, γ) is also an upward planar drawing of $\overrightarrow{G}$, by definition.

We prove the sufficiency. A directed plane graph $\overrightarrow{G}$ is called *plane st-digraph* if it has exactly one source s and one sink t such that s and t are both incident to the outer face of $\overrightarrow{G}$. Each face f of a plane st-digraph consists of two monotone paths called *left path* and *right path*, where the left path has f to the right when traversing it from its source to its sink. The right path and the left path of the outer face of $\overrightarrow{G}_{st}$ are also called *leftmost path* and *rightmost path* of $\overrightarrow{G}_{st}$, respectively.

Since $\overrightarrow{G}$ is upward planar, $\overrightarrow{G}$ can be augmented [DT88] to a plane st-digraph $\overrightarrow{G}_{st}$. Also, this can be done by adding only *dummy* edges (u, v) such that u and v are incident to the same face f, and u and v are either both sources or both sinks in C_f (when C_f is oriented according to $\overrightarrow{G}$). Note that, since (G, γ) is jagged, each dummy edge (u, v) is such that $\gamma(u) = \gamma(v)$.

We now compute the *directed dual* $\overrightarrow{G}_{s^*t^*}$ of $\overrightarrow{G}_{st}$. The vertices of $\overrightarrow{G}_{s^*t^*}$ are the faces of $\overrightarrow{G}_{st}$; two special vertices s^* and t^* represent the outer face. There is an edge (f, g) in $\overrightarrow{G}_{s^*t^*}$ if face f shares an edge $(u, v) \neq (s, t)$ with face g, and face f is on the left side of (u, v) when such an edge is traversed from u to v. Graph $\overrightarrow{G}_{s^*t^*}$ is a plane st-digraph [DT88].

We divide the plane into k horizontal strips of fixed height, each corresponding to one of the strips of (G, γ).

We compute an upward planar drawing of $\overrightarrow{G}_{st}$ as follows. First, consider the leftmost path p_l of $\overrightarrow{G}_{st}$, where $p_l = (s = v_1^1, \ldots, v_1^{h(1)}, v_2^1, \ldots, v_2^{h(2)}, \ldots, v_k^1, \ldots, v_k^{h(k)} =$

t), with $\gamma(v_i^1) = \cdots = \gamma(v_i^{h(i)}) = i$, for $i = 1, \ldots, k$. Path p_l is drawn as a y-monotone curve in which each vertex $u \in p_l$ lies inside strip $\gamma(u)$. Then, we add the faces of $\overrightarrow{G}_{st}$ one at a time, in such a way that a face is considered after all its predecessors in $\overrightarrow{G}_{s^*t^*}$. When a face f is considered, its left path has been already drawn as a y-monotone curve. We draw the right path of f as a y-monotone curve in which each vertex u lies inside strip $\gamma(u)$. This implies that the rightmost path of the graph in the current drawing is represented by a y-monotone curve.

A strip planar drawing of (G, γ) can be obtained from the upward planar drawing of $\overrightarrow{G}_{st}$ by removing the dummy edges. □

Note how the correctness of the proof of Lemma 6.6 heavily depends on the fact that (G, γ) is jagged. We thus obtain the main result of this chapter:

Theorem 6.1 *The* STRIP PLANARITY *testing problem can be solved in $O(|G|^2)$ time for instances (G, γ) such that G is a plane graph.*

Proof: In the following we denote by $|H|$ the number of vertices of any instance (H, γ) of STRIP PLANARITY; also, we denote by $r(H)$ the number of minima and maxima of (H, γ), and by $k(H)$ the number of strips of (H, γ). Further, we assume that $k(H) \leq |H|$, since empty strips can be removed without loss of generality.

Let (G, γ) be any instance of STRIP PLANARITY such that G is a plane graph.

By Lemma 6.1, there exists an $O(|G|^2)$-time algorithm that either decides that (G, γ) is not strip planar, or constructs a set $S = \{(G_1^*, \gamma_1^*), \ldots, (G_m^*, \gamma_m^*)\}$ of strict instances such that:

- (G, γ) is strip planar if and only if all of $(G_1^*, \gamma_1^*), \ldots, (G_m^*, \gamma_m^*)$ are strip planar;
- $\sum_{i=1}^{m} |G_i^*| \in O(|G|)$;
- $k(G_i^*) \leq k(G) + 1$, for each $1 \leq i \leq m$; and
- $\sum_{i=1}^{m} r(G_i^*) \in O(r(G))$.

Hence, it suffices to show that the STRIP PLANARITY of each (G_i^*, γ_i^*) can be tested in $O(|G_i^*|^2)$ time.

By Lemma 6.2, there exists an $O(k(G_i^*)|G_i^*|) \in O(|G_i^*|^2)$-time algorithm that constructs a strict proper instance (G_i^1, γ_i^1) that is equivalent to (G_i^*, γ_i^*), with $|G_i^1| \in O(k(G_i^*)|G_i^*|)$ vertices, with $k(G_i^1) = k(G_i^*)$, and with $r(G_i^1) = r(G_i^*)$.

By Lemma 6.3, there exists an $O(|G_i^1|) = O(k(G_i^*)|G_i^*|) \in O(|G_i^*|^2)$-time algorithm that constructs a 2-connected strict proper instance (G_i^2, γ_i^2) that is equivalent

to (G_i^1, γ_i^1), with $|G_i^2| \in O(|G_i^1|) \in O(k(G_i^*)|G_i^*|)$ vertices, with $k(G_i^2) = k(G_i^1) = k(G_i^*)$, and with $r(G_i^2) \in O(r(G_i^1)) \in O(r(G_i^*))$.

By Lemma 6.4, there exists an $O(k(G_i^2)r(G_i^2) + |G_i^2|) \in O(k(G_i^*)r(G_i^*) + k(G_i^*)|G_i^*|) \in O(|G_i^*|^2)$-time algorithm that constructs a quasi-jagged instance (G_i^3, γ_i^3) that is equivalent to (G_i^2, γ_i^2), with $|G_i^3| \in O(k(G_i^2)r(G_i^2) + |G_i^2|) \in O(k(G_i^*)|G_i^*|)$ vertices, with $k(G_i^3) = k(G_i^2) = k(G_i^*)$, and with $r(G_i^3) \in O(r(G_i^2)) \in O(r(G_i^*))$.

By Lemma 6.5, there exists an $O(k(G_i^3)r(G_i^3) + |G_i^3|) \in O(k(G_i^*)r(G_i^*) + k(G_i^*)|G_i^*|) \in O(|G_i^*|^2)$-time algorithm that constructs a jagged instance (G_i^4, γ_i^4) that is equivalent to (G_i^3, γ_i^3), with $|G_i^4| \in O(k(G_i^3)r(G_i^3) + |G_i^3|) \in O(k(G_i^*)|G_i^*|)$ vertices, with $k(G_i^4) = k(G_i^3) = k(G_i^*)$, and with $r(G_i^4) \in O(r(G_i^3)) \in O(r(G_i^*))$.

By Lemma 6.6, (G_i^4, γ_i^4) is strip planar if and only if the associated directed graph $\overrightarrow{G_i^4}$ of (G_i^4, γ_i^4) is upward planar. Observe that $\overrightarrow{G_i^4}$ can be constructed from (G_i^4, γ_i^4) in $O(|G_i^4|) = O(k(G_i^*)|G_i^*|) \in O(|G_i^*|^2)$ time.

Finally, by the results of Bertolazzi et al. [BDLM94], the UPWARD PLANARITY of $\overrightarrow{G_i^4}$ can be tested in $O(|G_i^4| + (s(G_i^4))^2)$ time, where $s(G_i^4)$ is the total number of sources and sinks of $\overrightarrow{G_i^4}$. Observe that $s(G_i^4) \leq r(G_i^4)$, hence the UPWARD PLANARITY of $\overrightarrow{G_i^4}$ can be tested in $O(|G_i^*|^2 + (r(G_i^*))^2) \in O(|G_i^*|^2)$ time.

This concludes the proof of the theorem. □

Non-Connected Instances

In this section we show how the problem of testing the STRIP PLANARITY of non-connected instances can be reduced to the one of testing the STRIP PLANARITY of connected instances.

The input of the STRIP PLANARITY testing algorithm might or might not specify the containment relationships between distinct connected components. In the case in which such relationships are not prescribed, a non-connected instance (G, γ) is strip planar if and only if all its connected components are strip planar. Namely, if (G, γ) is strip planar, then all its components are strip planar. Conversely, if all the components of (G, γ) are strip planar, then a strip planar drawing of (G, γ) can be obtained by placing strip planar drawings of the components of (G, γ) "side by side".

Assume now that the input (G, γ) of the STRIP PLANARITY testing algorithm specifies the containment relationships between distinct connected components. That is, the boundary of each face of G is prescribed by the input. Test individually the STRIP PLANARITY of each connected component of (G, γ). If one of the tests fails, then (G, γ) is not strip planar. Otherwise, construct a strip planar drawing of each connected component of (G, γ). Place the drawings of the connected components

containing edges incident to the outer face of G side by side. Repeatedly insert connected components in the internal faces of the currently drawn graph (G', γ) as follows. If a connected component (G_i, γ) of (G, γ) has to be placed inside an internal face f of (G', γ), check whether $\gamma(u_M) \leq \gamma(u_M^f)$ and whether $\gamma(u_m) \geq \gamma(u_m^f)$, where u_M (u_m) is a vertex of (G_i, γ) such that $\gamma(u_M)$ is maximum (resp. $\gamma(u_m)$ is minimum) among the vertices of G_i, and where u_M^f (u_m^f) is a vertex of C_f such that $\gamma(u_M^f)$ is maximum (resp. $\gamma(u_m^f)$ is minimum) among the vertices of C_f. If the test fails, then (G, γ) is not strip planar. Otherwise, using a technique analogous to the one of Claim 6.4, a strip planar drawing of (G', γ) can be modified so that two consecutive global minimum and maximum for f can be connected by a y-monotone curve $\mathcal{C}$ inside f. Suitably squeezing a strip planar drawing of (G_i, γ) and placing it arbitrarily close to $\mathcal{C}$ provides a strip planar drawing of $(G' \cup G_i, \gamma)$. Repeating such an argument leads either to conclude that (G, γ) is not strip planar, or to construct a strip planar drawing of (G, γ).

6.4 Reduction

In this section we show that the STRIP PLANARITY testing problem reduces in polynomial time to the C-PLANARITY testing problem.

Theorem 6.2 *Let (G, γ) be an instance of* STRIP PLANARITY. *Then, there exists an instance $C(G', T)$ of* C-PLANARITY *such that (G, γ) is strip planar if and only if $C(G', T)$ is clustered planar. Further, $C(G', T)$ can be constructed in polynomial time.*

Proof: Denote by k the number of strips of (G, γ). First, we show that, if $k \leq 2$, then the "natural" reduction from STRIP PLANARITY to C-PLANARITY, namely the one that transforms each strip into a cluster, is a valid polynomial-time reduction. We now formalize this claim.

If $k = 1$, clustered graph $C(G', T)$ is defined as follows. Graph G' coincides with G and tree T consists of a single internal node μ that is parent of all the vertices of G'. The equivalence between the STRIP PLANARITY of (G, γ) and the C-PLANARITY of $C(G', T)$ follows from their equivalence to the planarity of $G = G'$.

If $k = 2$, clustered graph $C(G', T)$ is defined as follows. Graph G' coincides with G and tree T consists of three internal nodes μ, μ_1, and μ_2, where μ is parent of μ_1 and μ_2, and where μ_i is parent of every vertex x of G' such $\gamma(x) = i$, for $i = 1, 2$. From a strip planar drawing Γ of (G, γ), a c-planar drawing Γ' of $C(G', T)$ can be constructed so that the drawings of G and G' coincide, and so that, for $i = 1, 2$, the

region $R(\mu_i)$ representing μ_i is a rectangle whose top and bottom sides lie on the top and bottom lines delimiting γ_i, respectively, and whose left (right) side is to the left (right) of all the vertices and edges of G'. Conversely, suppose that $C(G', T)$ is c-planar. Then, it admits a c-planar straight-line drawing Γ' in which μ_1 and μ_2 are represented by convex regions $R(\mu_1)$ and $R(\mu_2)$ (see [AFK11, EFLN06]). Thus, $R(\mu_1)$ and $R(\mu_2)$ can be separated by a straight line l; by suitably rotating l and the Cartesian axes, we can assume that l is horizontal and every edge of G' is y-monotone in Γ', with $R(\mu_1)$ below $R(\mu_2)$. Then, define γ_1 (γ_2) as a horizontal strip containing $R(\mu_1)$ (resp. $R(\mu_2)$) and entirely below l (resp. above l). The resulting drawing Γ is a strip planar drawing of (G, γ).

If $k \geq 3$, then the above reduction does not always work (Fig.6.1 shows an example with $k = 4$). In the following we show how to construct a clustered graph $C(G', T)$ whose C-PLANARITY is equivalent to the STRIP PLANARITY of (G, γ) if $k \geq 3$. We also assume that G is connected. This is not a loss of generality. Namely, if G is not connected, then (G, γ) is strip planar if and only if each of its connected components $(G_1, \gamma_1), \ldots, (G_m, \gamma_m)$ is strip planar (where $\gamma_i(v) = \gamma(v)$ for every $v \in G_i$ and every $1 \leq i \leq m$). Thus, if an instance $C_i(G'_i, T_i)$ can be constructed in polynomial time equivalent to (G_i, γ_i), for every $1 \leq i \leq m$, then an instance $C(G', T)$ can also be constructed in polynomial time equivalent to (G, γ), where $G' = G'_1 \cup \cdots \cup G'_m$, and where T is a tree consisting of a root having $T_1, \ldots, T_m$ as subtrees.

Further, we assume that (G, γ) is proper. If this is not the case, then the reduction described in Section 6.3 can be applied in order to obtain an equivalent proper instance.

Summarizing, we can suppose w.l.o.g. that (G, γ) is connected, proper, and has $k \geq 3$ strips. We now describe how to construct $C(G', T)$ (see Figs. 6.20(a)-(b)).

Graph G' is composed of G and of a triconnected plane graph H, which consists of vertices a, b, c, d, $u_1, \ldots, u_k$, $v_1, \ldots, v_k$, and of edges (a, b), (b, c), (c, d), (a, d), (b, d), (a, u_k), (a, v_k), (c, u_1), (c, v_1), $(b, u_1), \ldots, (b, u_k)$, $(d, v_1), \ldots, (d, v_k)$, $(u_1, u_2), \ldots, (u_{k-1}, u_k)$, and $(v_1, v_2), \ldots, (v_{k-1}, v_k)$.

Tree T is constructed as follows. Initialize T with a root cluster μ. Add to T four clusters μ_a, μ_b, μ_c, and μ_d as children of μ, containing vertices a, b, c, and d, respectively. Then, for each $i = 1, \ldots, k$, add a cluster μ_i to T, as a child of μ, that contains vertices u_i, v_i, and each vertex $w \in G$ such that $\gamma(w) = i$.

Clearly, $C(G', T)$ can be constructed in polynomial time. We prove that $C(G', T)$ admits a c-planar drawing if and only if (G, γ) admits a strip planar drawing.

Suppose that $C(G', T)$ admits a c-planar drawing Γ'. We construct a strip planar drawing Γ of (G, γ) as follows.

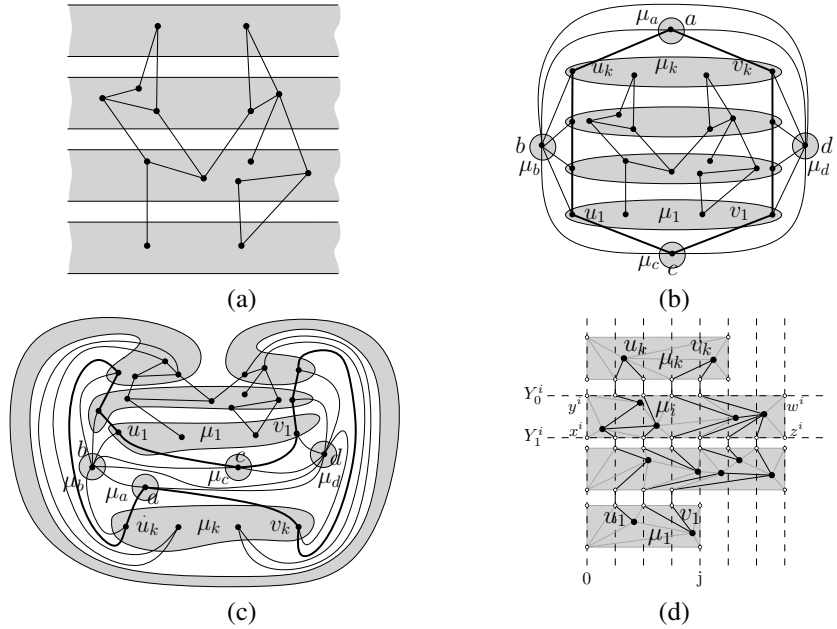


Figure 6.20: Illustration for the proof of Theorem 6.2. (a) Instance (G, γ) of STRIP PLANARITY; (b) instance $C(G', T)$ of C-PLANARITY obtained from (G, γ); (c) a c-planar drawing Γ' of $C(G', T)$; (d) the strip planar drawing Γ of (G, γ) obtained from Γ'.

Since H is triconnected, it has a unique planar embedding. Moreover, since $k \geq 3$, just one of the faces of H has incident vertices belonging to all clusters $\mu_1, \ldots, \mu_k$, namely the face f delimited by cycle $C_f = (u_1, \ldots, u_k, a, v_k, \ldots, v_1, c)$. Hence, all the vertices and edges of G are embedded inside f in Γ', given that G is connected. Moreover, for each $1 \leq i \leq k$, the intersection of the region $R(\mu_i)$ representing cluster μ_i in Γ' with the interior of f is a connected region containing u_i and v_i, given that u_i and v_i are separated by path (a, b, c) in the region delimited by cycle C_f different from f. Then, the edges connecting a vertex of μ_i to a vertex of μ_{i+1} cut the boundary of $R(\mu_i)$ consecutively, for every $1 \leq i \leq k-1$; denote by $s^i_1, \ldots, s^i_{n_i}$ the clockwise order in which these edges cut the boundary of $R(\mu_i)$, starting at the first edge s^i_1 crossing the boundary of $R(\mu_i)$ after (u_i, u_{i+1}). Analogously, the m_i edges

connecting a vertex of μ_i to a vertex of μ_{i-1} cut the boundary of $R(\mu_i)$ consecutively, for every $2 \leq i \leq k$. Observe that, since (G, γ) is proper, it holds that $n_i = m_{i+1}$ for each $i = 1, \ldots, k-1$.

We now show how to construct Γ. We first construct an auxiliary graph C_x. Initialize C_x as G. We replace each edge s_j^i, where s_j^i connects a vertex u_j^i in μ_i to a vertex u_j^{i+1} in μ_{i+1}, with a path $(u_j^i, p_j^i, q_j^{i+1}, u_j^{i+1})$. Further, add to C_x (i) dummy edges (p_j^i, p_{j+1}^i), with $j = 1, \ldots, n_i - 1$, (ii) dummy edges (q_j^i, q_{j+1}^i), with $j = 1, \ldots, m_i - 1$, (iii) dummy edges (p_1^i, y^i), (x^i, y^i), and (x^i, q_1^i), where x^i and y^i are two dummy vertices, and (iv) dummy edges $(p_{n_i}^i, w^i)$, (z^i, w^i), and $(z^i, q_{m_i}^i)$, where z^i and w^i are two dummy vertices. Also, add edges (x^1, z^1) and (y^k, w^k) to C_x.

For each $i = 1, \ldots, k$, denote by C_{μ_i} the subgraph of C_x induced by the vertices inside or on the boundary of cycle $\mathcal{C}^i = (p_1^i, \ldots, p_n^i, w^i, z^i, q_{m_i}^i, \ldots, q_1^i, x^i, y^i)$.

Consider any set of k horizontal strips $\gamma_1, \ldots, \gamma_k$. For each $i = 1, \ldots, k$, let $y = Y_0^i$ and $y = Y_1^i$ be the higher and lower horizontal lines delimiting the strip γ_i, respectively.

We first show how to draw each graph C_{μ_i}. Place vertex y^i at point $(0, Y_0^i)$, vertex x^i at point $(0, Y_1^i)$, vertex w^i at point $(\max\{m_i, n_i\} + 1, Y_0^i)$, and vertex z^i at point $(\max\{m_i, n_i\} + 1, Y_1^i)$. Also, place each vertex p_j^i at point (j, Y_0^i), and each vertex q_j^i at point (j, Y_1^i). By construction, $\mathcal{C}^i$ is represented by a convex quadrilateral Q^i. Then, extend Q^i to a straight-line planar drawing Γ_i of C_{μ_i}. Observe that C_{μ_i} can be augmented to an internally-triangulated planar graph with no edge connecting two non-consecutive vertices on the outer face. Hence, Γ_i always exists [CEGL12]. Slightly perturbing the position of the internal vertices of C_{μ_i} results in a drawing in which all the edges, except for the ones incident to the outer face, are y-monotone.

Finally, for each $i = 1, \ldots, k-1$ and $j = 1, \ldots, n_i$, vertices p_j^i and q_j^{i+1} have the same x-coordinate, and hence can be connected with a vertical straight-line segment not intersecting any other edge. Now removing the inserted dummy edges and replacing all dummy vertices p_j^i and q_j^i with bends results in a strip planar drawing Γ of (G, γ).

Suppose now that (G, γ) admits a strip planar drawing Γ. A c-planar drawing Γ' of $C(G', T)$ can be constructed as follows. First, let the drawings of G' in Γ' and of G in Γ coincide. Let X_0 and X_1 be the smallest and the largest x-coordinate of a vertex in Γ, respectively. For each $i = 1, \ldots, k$, let $y = Y_0^i$ and $y = Y_1^i$ be the horizontal lines delimiting strip γ_i from above and from below, respectively. Refer to Fig. 6.21.

Place vertices u_i and v_i at points $(X_0 - 1, \frac{Y_0^i + Y_1^i}{2})$ and $(X_1 + 1, \frac{Y_0^i + Y_1^i}{2})$, respectively, and represent μ_i as a rectangular region $R(\mu_i)$ with corners at $(X_0 - 2, Y_0^i)$, $(X_0 - 2, Y_1^i)$, $(X_1 + 2, Y_0^i)$, and $(X_1 + 2, Y_1^i)$. Then, place vertex a at point $(X_a =$

$\frac{X_0+X_1}{2}, Y_a = Y_0^k+1)$, vertex b at point $(X_b = X_0-4, Y_b = \frac{Y_0^k+Y_1^1}{2})$, vertex c at point $(X_c = \frac{X_0+X_1}{2}, Y_c = Y_1^1 - 1)$, and vertex d at point $(X_d = X_1 + 4, Y_d = \frac{Y_0^k+Y_1^1}{2})$.

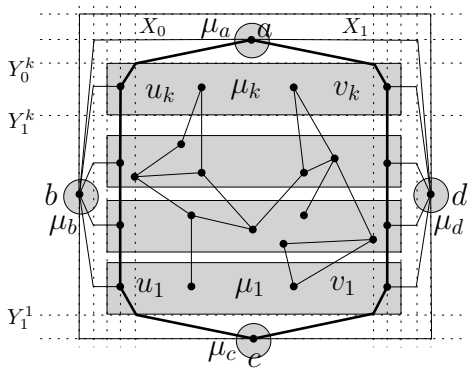


Figure 6.21: The c-planar drawing Γ' of $C(G', T)$ obtained from a strip planar drawing Γ of (G, γ).

Draw edges (a, b), (b, c), (c, d), and (d, a) as polygonal lines bending at points (X_b+1, Y_a), (X_b, Y_c), (X_d, Y_c), and (X_d-1, Y_a), respectively. Draw edge (b, d) as a polygonal line bending at points (X_b, Y_a+1) and (X_d, Y_a+1). For each $i = 1, \ldots, k$, draw edges (b, u_i) and (d, v_i) as polygonal lines bending at points $(X_b + 1, \frac{Y_0^i+Y_1^i}{2})$ and $(X_d - 1, \frac{Y_0^i+Y_1^i}{2})$, respectively. Draw edges (a, u_k), (a, v_k), (c, u_1), and (c, v_1) as polygonal lines bending at points (X_0, Y_0^k), (X_1, Y_0^k), (X_0, Y_1^1), and (X_1, Y_1^1), respectively. Draw edges (u_i, u_{i+1}) and (v_i, v_{i+1}) as straight-line segments, for each $1 \leq i \leq k - 1$.

Finally, draw cluster μ_a (μ_b, μ_c, μ_d) as a small disk $R(\mu_a)$ (resp. $R(\mu_b)$, $R(\mu_c)$, $R(\mu_d)$) enclosing only vertex a (resp. b, c, d) and not overlapping with any other region.

This results in a c-planar drawing of $C(G', T)$. □

6.5 Conclusions

In this chapter, we introduced the STRIP PLANARITY testing problem and showed how to solve it in polynomial time if the input graph has a prescribed plane embedding. The main question raised by this chapter is whether the STRIP PLANARITY testing problem can be solved in polynomial time or it is rather $\mathcal{NP}$-hard for graphs without

a prescribed plane embedding. The problem is intriguing even if the input graph is a tree.

We also proved the existence of a polynomial-time reduction from the STRIP PLANARITY testing problem to the C-PLANARITY testing problem. Recently, Fulek proved [Ful14] a stronger result: For every instance (G, γ) of STRIP PLANARITY, an equivalent instance $C(G, T)$ of C-PLANARITY can be constructed in polynomial time such that T only contains three clusters. Thus, designing a polynomial-time algorithm for the STRIP PLANARITY testing problem is a vital step towards deepening our understanding of C-PLANARITY.

Chapter 7

C-Level Planarity and T-Level Planarity Testing

In this chapter[1] we study two problems related to the drawing of level graphs and clustered graphs, that is, T-LEVEL PLANARITY and CLUSTERED-LEVEL PLANARITY. We show that both problems are NP-complete in the general case and that they become polynomial-time solvable when restricted to proper instances.

7.1 Introduction and Overview

A level graph is *proper* if every of its edges spans just two consecutive levels. Several papers dealing with the construction of level drawings of level graphs assume that the input graph is proper. Otherwise, they suggest to make it proper by "simply adding dummy vertices" along the edges spanning more than two levels. In this chapter we show that this apparently innocent augmentation has dramatic consequences if, instead of constructing just a level drawing, we are also interested in representing additional constraints, like a clustering of the vertices or consecutivity constraints on the orderings of the vertices along the levels.

A *level graph* $G = (V, E, \gamma)$ is a graph with a function $\gamma : V \to \{1, 2, ..., k\}$, with $1 \leq k \leq |V|$ such that $\gamma(u) \neq \gamma(v)$ for each edge $(u, v) \in E$. The set $V_i = \{v \mid \gamma(v) = i\}$ is the i-th *level* of G. A level graph $G = (V, E, \gamma)$ is *proper* if for every edge $(u, v) \in E$, it holds $\gamma(u) = \gamma(v) \pm 1$. A *level planar drawing* of (V, E, γ) maps

[1]The contents of this chapter are a joint work with Patrizio Angelini, Giuseppe Di Battista, Fabrizio Frati and Vincenzo Roselli, appeared partially in [ALD+14b] and in a journal [ALD+15]. Thanks to Maurizio Patrignani for fruitful discussions and helpful comments.

each vertex v of each level V_i to a point on the line $y = i$, denoted by L_i, and each edge to a y-monotone curve between its endpoints so that no two edges intersect. A level graph is *level planar* if it admits a level planar drawing. A linear-time algorithm for testing level planarity was presented by Jünger *et al.* [JLM98].

A *clustered-level graph* (*cl-graph*) (V, E, γ, T) is a level graph (V, E, γ) equipped with a *cluster hierarchy* T, that is, a rooted tree where each leaf is an element of V and each internal node μ, called *cluster*, represents the subset V_μ of V composed of the leaves of the subtree of T rooted at μ. A *clustered-level planar drawing* (*cl-planar* drawing) of (V, E, γ, T) is a level planar drawing of level graph (V, E, γ) together with a representation of each cluster μ as a simple closed region enclosing all and only the vertices in V_μ such that: (1) no edge intersects the boundary of a cluster more than once; (2) no two cluster boundaries intersect; and (3) the intersection of L_i with any cluster μ is a straight-line segment, that is, the vertices of V_i that belong to μ are consecutive along L_i. A cl-graph is *clustered-level planar* (*cl-planar*) if it admits a cl-planar drawing. CLUSTERED-LEVEL PLANARITY (CL-PLANARITY) is the problem of testing whether a given cl-graph is cl-planar. This problem was introduced by Forster and Bachmaier [FB04], who showed a polynomial-time testing algorithm for the case in which the level graph is a proper hierarchy and the clusters are level-connected.

A *$\mathcal{T}$-level graph* (also known as *generalized k-ary tanglegram*) $(V, E, \gamma, \mathcal{T})$ is a level graph (V, E, γ) equipped with a set $\mathcal{T} = \{T_1, \ldots, T_k\}$ of trees such that the leaves of T_i are the vertices of level V_i of (V, E, γ), for $1 \leq i \leq k$. A *$\mathcal{T}$-level planar drawing* of $(V, E, \gamma, \mathcal{T})$ is a level planar drawing of (V, E, γ) such that, for $i = 1, \ldots, k$, the order in which the vertices of V_i appear along L_i is *compatible* with T_i, that is, for each node w of T_i, the leaves of the subtree of T_i rooted at w appear consecutively along L_i. A $\mathcal{T}$-level graph is *$\mathcal{T}$-level planar* if it admits a $\mathcal{T}$-level planar drawing. T-LEVEL PLANARITY is the problem of testing whether a given $\mathcal{T}$-level graph is $\mathcal{T}$-level planar. This problem was introduced by Wotzlaw *et al.* [WSP12], who showed a quadratic-time algorithm for the case in which the number of vertices in each level is constant.

The definition of *proper* naturally extends to cl-graphs and $\mathcal{T}$-level graphs. Note that, given any non-proper level graph G it is easy to construct a proper level graph G' that is level planar if and only if G is level planar. However, as mentioned above, there exists no trivial transformation from a non-proper cl-graph (a non-proper $\mathcal{T}$-level graph) to an equivalent proper cl-graph (resp., an equivalent proper $\mathcal{T}$-level graph).

In this chapter we show that CL-PLANARITY and T-LEVEL PLANARITY are NP-complete for non-proper instances. Conversely, we show that both problems are polynomial-time solvable for proper instances.

Our results have several consequences: (1) They narrow the gap between polyno-

miality and NP-completeness in the classification of Schaefer [Sch13] (see Fig. 0.1 in the Introduction and Fig. 7.1 in which the contributions of the chapters of Parts II and III are highlighted). Note that the reduction of Schaefer between T-LEVEL PLANARITY and SEFE-2 holds for proper instances [Sch13]. (2) They allow to partially answer a question from [Sch13] asking whether a reduction exists from CL-PLANARITY to SEFE-2. We show that such a reduction exists for proper instances and that a reduction from general instances would imply the NP-hardness of SEFE-2. (3) They improve on the results of Forster and Bachmaier [FB04] and of Wotzlaw *et al.* [WSP12] by extending the classes of instances which are decidable in polynomial-time for CL-PLANARITY and T-LEVEL PLANARITY, respectively. (4) They provide the first, as far as we know, NP-completeness for a problem that has all the constraints of the clustered planarity problem (and some more).

The chapter is organized as follows. The NP-completeness proofs are in Section 7.2, while the algorithms are in Section 7.3. We conclude with open problems in Section 7.4.

7.2 NP-Hardness

In this section we prove that the T-LEVEL PLANARITY and the CL-PLANARITY problems are NP-complete. In both cases, the NP-hardness is proved by means of a polynomial-time reduction from the NP-complete problem BETWEENNESS [Opa79], that takes as input a finite set A of n objects and a set C of m ordered triples of distinct elements of A, and asks whether a linear ordering $\mathcal{O}$ of the elements of A exists such that for each triple $\langle \alpha, \beta, \delta \rangle$ of C, we have either $\mathcal{O} = <\ldots, \alpha, \ldots, \beta, \ldots, \delta, \ldots>$ or $\mathcal{O} = <\ldots, \delta, \ldots, \beta, \ldots, \alpha, \ldots>$.

Theorem 7.1 *T-LEVEL PLANARITY is NP-complete.*

Proof: The problem clearly belongs to NP. We prove the NP-hardness. Given an instance $\langle A, C \rangle$ of BETWEENNESS, we construct an equivalent instance $(V, E, \gamma, \mathcal{T})$ of T-LEVEL PLANARITY as follows. Let $A = \{1, \ldots, n\}$ and $m = |C|$. The graph (V, E) is a tree composed of n paths all incident to a common vertex v. More in detail, (V, E) is constructed as follows. Refer to Fig. 7.2(a).

Initialize $V = \{v\}$ and $E = \emptyset$, set $\gamma(v) = 0$, and let $T_0 \in \mathcal{T}$ be a tree with a single node v. For each $j = 1, \ldots, n$, add a vertex v_j to V, with $\gamma(v_j) = 1$, and add an edge (v, v_j) to E. Also, initialize a variable $last(j) = v_j$. The tree $T_1 \in \mathcal{T}$ is a star whose leaves are all the vertices of the level V_1.

Then, for each $i = 1, \ldots, m$, consider the triple $t_i = \langle \alpha, \beta, \delta \rangle$ of C. Add six vertices $u_\alpha(i)$, $u'_\alpha(i)$, $u_\beta(i)$, $u'_\beta(i)$, $u_\delta(i)$, and $u'_\delta(i)$ to V, with $\gamma(u_\alpha(i)) = \gamma(u_\beta(i)) =$

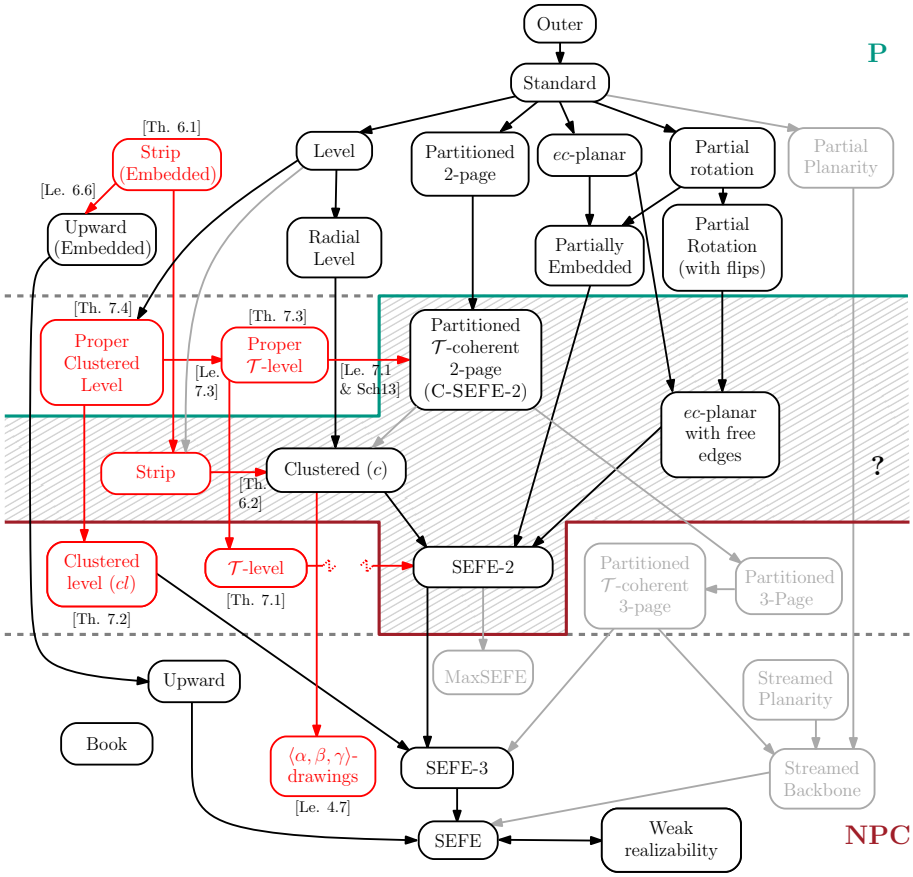


Figure 7.1: View of the schema proposed in the Introduction in which the contributions of Chapters 4 and 5 of Part II, and the contributions of Chapters 6 and 7 of Part III are highlighted by using the red color. The prefix "proper" has been added to two classes in [Sch13] to better clarify their nature.

$\gamma(u_\delta(i)) = 2i$ and $\gamma(u'_\alpha(i)) = \gamma(u'_\beta(i)) = \gamma(u'_\delta(i)) = 2i + 1$. Also, add six edges $(last(\alpha), u_\alpha(i))$, $(last(\beta), u_\beta(i))$, $(last(\delta), u_\delta(i))$, $(u_\alpha(i), u'_\alpha(i))$, $(u_\beta(i), u'_\beta(i))$, and $(u_\delta(i), u'_\delta(i))$ to E. Further, set $last(\alpha) = u'_\alpha(i)$, $last(\beta) = u'_\beta(i)$, and $last(\delta) = u'_\delta(i)$. Let $T_{2i} \in \mathcal{T}$ be a binary tree with a root r_{2i}, an internal node x_{2i} and a leaf $u_\alpha(i)$ both adjacent to r_{2i}, and with leaves $u_\beta(i)$ and $u_\delta(i)$ both adjacent to x_{2i}.

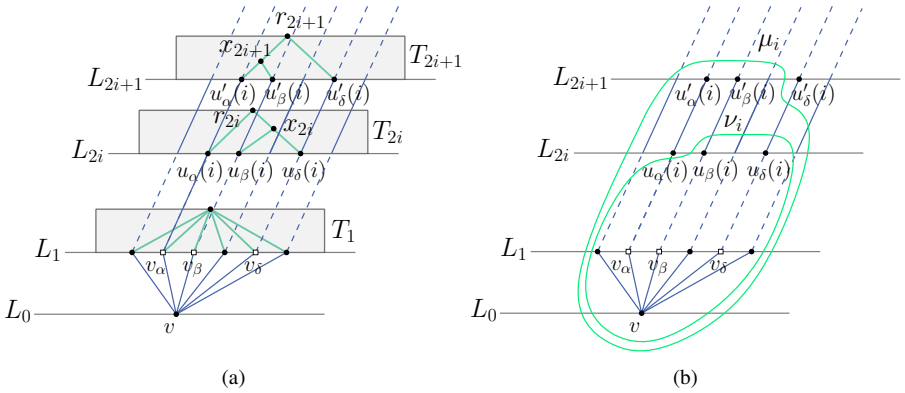


Figure 7.2: Illustrations for the proof of (a) Theorem 7.1 and (b) Theorem 7.2.

Moreover, let $T_{2i+1} \in \mathcal{T}$ be a binary tree with a root r_{2i+1}, an internal node x_{2i+1} and a leaf $u'_\delta(i)$ both adjacent to r_{2i+1}, and two leaves $u'_\alpha(i)$ and $u'_\beta(i)$ both adjacent to x_{2i+1}.

The reduction is easily performed in $O(n+m)$ time. We prove that $(V, E, \gamma, \mathcal{T})$ is $\mathcal{T}$-level planar if and only if $\langle A, C\rangle$ is a positive instance of BETWEENNESS.

Suppose that $(V, E, \gamma, \mathcal{T})$ admits a $\mathcal{T}$-level planar drawing Γ. Consider the left-to-right order $\mathcal{O}_1$ in which the vertices of the level V_1 appear along L_1. Construct an order $\mathcal{O}$ of the elements of A such that $\alpha \in A$ appears before $\beta \in A$ if and only if $v_\alpha \in V_1$ appears before $v_\beta \in V_1$ in $\mathcal{O}_1$. In order to prove that $\mathcal{O}$ is a positive solution for $\langle A, C\rangle$ it suffices to prove that, for each triple $t_i = \langle \alpha, \beta, \delta\rangle \in C$, vertices v_α, v_β, and v_δ appear either in this order or in the reverse order in $\mathcal{O}_1$. Note that the tree T_{2i} enforces $u_\alpha(i)$ not to lie between $u_\beta(i)$ and $u_\delta(i)$ along L_{2i}; also, the tree T_{2i+1} enforces $u'_\delta(i)$ not to lie between $u'_\alpha(i)$ and $u'_\beta(i)$ along L_{2i+1}. Since the three paths connecting $u'_\alpha(i)$, $u'_\beta(i)$, and $u'_\delta(i)$ with v are y-monotone, do not cross each other, and contain $u_\alpha(i)$ and v_α, $u_\beta(i)$ and v_β, and $u_\delta(i)$ and v_δ, respectively, we have that v_α, v_β, and v_δ appear either in this order or in the reverse order in $\mathcal{O}_1$.

Suppose that an ordering $\mathcal{O}$ of the elements of A exists that is a positive solution of BETWEENNESS for the instance $\langle A, C\rangle$. In order to construct Γ, place the vertices of V_1 along L_1 in such a way that each vertex $v_j \in V_1$, for $j = 1, \ldots, n$, is assigned x-coordinate equal to s if j is the s-th element of $\mathcal{O}$. Also, for $i = 1, \ldots, m$, consider the triple $t_i = \langle \alpha, \beta, \delta\rangle \in C$. Place the vertices $u_\lambda(i)$ and $u'_\lambda(i)$, with $\lambda \in \{\alpha, \beta, \delta\}$, on L_{2i} and L_{2i+1}, respectively, in such a way that $u_\lambda(i)$ and $u'_\lambda(i)$ are assigned x-

coordinate equal to s if λ is the s-th element of $\mathcal{O}$. Finally, place v at any point on L_0 and draw the edges of E as straight-line segments. We prove that Γ is a $\mathcal{T}$-level planar drawing of $(V, E, \gamma, \mathcal{T})$. First, Γ is a level planar drawing of (V, E, γ), by construction. Further, for each $i = 1, \ldots, m$, the vertices $u_\alpha(i)$, $u_\beta(i)$, and $u_\delta(i)$ appear along L_{2i} either in this order or in the reverse order; in both cases, the order is compatible with the tree T_{2i}. Analogously, $u'_\alpha(i)$, $u'_\beta(i)$, and $u'_\delta(i)$ appear along L_{2i+1} either in this order or in the reverse order; in both cases, the order is compatible with T_{2i+1}. Finally, the order in which vertices of V_0 and V_1 appear along L_0 and L_1 are trivially compatible with T_0 and T_1, respectively. □

Theorem 7.2 CLUSTERED-LEVEL PLANARITY *is NP-complete.*

Proof: The problem clearly belongs to NP. We prove the NP-hardness. Given an instance $\langle A, C\rangle$ of BETWEENNESS, we construct an instance $(V, E, \gamma, \mathcal{T})$ of T-LEVEL PLANARITY as in the proof of Theorem 7.1; then, starting from $(V, E, \gamma, \mathcal{T})$, we construct an instance (V, E, γ, T) of CL-PLANARITY that is cl-planar if and only if $(V, E, \gamma, \mathcal{T})$ is $\mathcal{T}$-level planar. This, together with the fact that $(V, E, \gamma, \mathcal{T})$ is $\mathcal{T}$-level planar if and only if $\langle A, C\rangle$ is a positive instance of BETWEENNESS, implies the NP-hardness of CL-PLANARITY. Refer to Fig. 7.2(b).

We describe how to construct (V, E, γ, T) from $(V, E, \gamma, \mathcal{T})$ by defining the cluster hierarchy T as follows. Initialize T with a root ρ, and insert $u'_\delta(m)$ and a node μ_m as children of ρ. Then for $i = m, \ldots, 1$, perform the following insertions in T: Insert $u'_\alpha(i)$, $u'_\beta(i)$, $u_\alpha(i)$, and a node ν_i in T as children of μ_i; then insert $u_\beta(i)$, $u_\delta(i)$, x, and a node y in T as children of ν_i, where $x = u'_\delta(i-1)$ and $y = \mu_{i-1}$, if $i > 1$, and $x = V_0 \cup V_1$ and $y = \emptyset$, if $i = 1$.

We prove that (V, E, γ, T) is cl-planar if and only if $(V, E, \gamma, \mathcal{T})$ is $\mathcal{T}$-level planar.

Suppose that (V, E, γ, T) admits a cl-planar drawing Γ. Construct a $\mathcal{T}$-level planar drawing Γ^* of $(V, E, \gamma, \mathcal{T})$ by removing from Γ the clusters of T. The drawing of (V, E, γ) in Γ^* is level-planar, since it is level-planar in Γ. Further, for each $i = 1, \ldots, m$, the vertex $u_\alpha(i)$ does not appear between $u_\beta(i)$ and $u_\delta(i)$ along L_{2i}, since $u_\beta(i), u_\delta(i) \in V_{\nu_i}$ and $u_\alpha(i) \notin V_{\nu_i}$; analogously, $u'_\delta(i)$ does not appear between $u'_\alpha(i)$ and $u'_\beta(i)$ along L_{2i+1}, since $u'_\alpha(i), u'_\beta(i) \in V_{\mu_i}$ and $u'_\delta(i) \notin V_{\mu_i}$. Hence, the order of the vertices of V_{2i} and V_{2i+1} along L_{2i} and L_{2i+1}, respectively, are compatible with the trees T_{2i} and T_{2i+1}. Finally, the order in which the vertices in V_0 and V_1 appear along L_0 and L_1 are trivially compatible with T_0 and T_1, respectively.

Suppose that $(V, E, \gamma, \mathcal{T})$ admits a $\mathcal{T}$-level planar drawing Γ^*; we describe how to construct a cl-planar drawing Γ of (V, E, γ, T). Assume that Γ^* is a straight-line drawing, which is not a loss of generality [EFLN06]. Initialize $\Gamma = \Gamma^*$. Draw

each cluster α in T as a convex region $R(\alpha)$ in Γ slightly surrounding the border of the convex hull of its vertices and slightly surrounding the border of the regions representing the clusters that are its descendants in T. Let j be the largest index such that V_j contains a vertex of α. Then, $R(\alpha)$ contains all and only the vertices that are descendants of α in T; moreover, any two clusters α and β in T are one contained into the other, hence $R(\alpha)$ and $R(\beta)$ do not cross; finally, we prove that no edge e in E crosses more than once the boundary of $R(\alpha)$ in Γ. First, if at least one end-vertex of e belongs to α, then e and the boundary of $R(\alpha)$ cross at most once, given that e is a straight-line segment and that $R(\alpha)$ is convex. All the vertices in $V_0 \cup \cdots \cup V_{j-1}$ and at least two vertices of V_j belong to α, hence their incident edges do not cross the boundary of $R(\alpha)$ more than once. Further, all the vertices in $V_{j+1} \cup \cdots \cup V_{2m+3}$ have y-coordinates larger than every point of $R(\alpha)$, hence edges between them do not cross $R(\alpha)$. It remains to consider the case in which e connects a vertex x_1 in V_j not in α (there is at most one such vertex) with a vertex x_2 in $V_{j+1} \cup \cdots \cup V_{2m+2}$; in this case e and $R(\alpha)$ do not cross given that x_1 is outside $R(\alpha)$, that x_2 has y-coordinate larger than every point of $R(\alpha)$, and that $R(\alpha)$ is arbitrarily close to the convex hull of its vertices. □

The reductions described in Theorems 7.1 and 7.2 can be modified so that (V, E) consists of a set of paths (by removing the levels V_0 and V_1), or so that (V, E) is a 2-connected series-parallel graph (by introducing two levels V_{2m+2} and V_{2m+3} "symmetric" to levels V_1 and V_0, respectively).

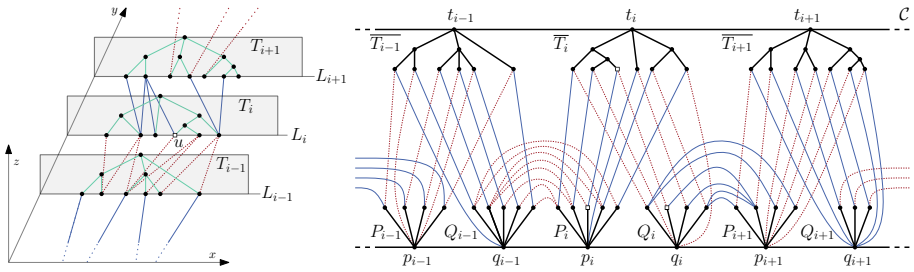
7.3 Polynomial-Time Algorithms

In this section we prove that both T-LEVEL PLANARITY and CL-PLANARITY are polynomial-time solvable problems if restricted to proper instances.

T-LEVEL PLANARITY

We start by describing a polynomial-time algorithm for T-LEVEL PLANARITY that is based on a reduction to the *Simultaneous Embedding with Fixed Edges* problem for two graphs (SEFE-2), which is defined as follows.

A *simultaneous embedding with fixed edges* (SEFE) of two graphs $G_1 = (V, E_1)$ and $G_2 = (V, E_2)$ on the same set V of vertices consists of two planar drawings Γ_1 and Γ_2 of G_1 and G_2, respectively, such that each vertex $v \in V$ is mapped to the same point in both drawings and each edge of the *common graph* $G_\cap = (V, E_1 \cap E_2)$ is represented by the same simple curve in the two drawings. The SEFE-2 problem asks whether a given pair of graphs $\langle G_1, G_2 \rangle$ admits a SEFE [BKR13b]. The computa-



(a) (b)

Figure 7.3: Illustration for the proof of Lemma 7.1. Index i is assumed to be even. (a) A T-level planar drawing Γ of instance $(V, E, \gamma, \mathcal{T})$. (b) The SEFE $\langle \Gamma_1, \Gamma_2 \rangle$ of the instance $\langle G_1^*, G_2^* \rangle$ of SEFE-2 corresponding to Γ. The correspondence between a vertex $u \in V_i$ and the leaves $u(\overline{T_i}) \in \overline{T_i}$, $u(P_i) \in P_i$, and $u(Q_i) \in Q_i$ is highlighted by representing all such vertices as white boxes.

tional complexity of the SEFE-2 problem is unknown, but there exist polynomial-time algorithms for instances that respect some conditions [ADF+12, BKR13b, BKR13a, BR13, Sch13]. We are going to use a result by Bläsius and Rütter [BR13], who proposed a quadratic-time algorithm for instances $\langle G_1, G_2 \rangle$ of SEFE-2 in which G_1 and G_2 are 2-connected and the common graph $G_\cap$ is connected.

In the following, by *size* of an instance $(V, E, \gamma, \mathcal{T})$ of T-LEVEL PLANARITY we mean the number of vertices in V plus the total number of internal nodes of the trees in $\mathcal{T}$; also, by *size* of an instance (V, E, γ, T) of CL-PLANARITY we mean the number of vertices in V plus the number of internal nodes of T.

Lemma 7.1 *Let $(V, E, \gamma, \mathcal{T})$ be a proper instance of T-LEVEL PLANARITY. There exists an equivalent instance $\langle G_1^*, G_2^* \rangle$ of SEFE-2 such that $G_1^* = (V^*, E_1^*)$ and $G_2^* = (V^*, E_2^*)$ are 2-connected and the common graph $G_\cap = (V^*, E_1^* \cap E_2^*)$ is connected. Further, the instance $\langle G_1^*, G_2^* \rangle$ can be constructed in linear time.*

Proof: We describe how to construct the instance $\langle G_1^*, G_2^* \rangle$. Refer to Fig. 7.3.

The common graph $G_\cap$ contains a cycle $\mathcal{C} = (t_1, t_2, \ldots, t_k, q_k, p_k, q_{k-1}, p_{k-1}, \ldots, q_1, p_1)$, where k is the number of levels of $(V, E, \gamma, \mathcal{T})$. For each $i = 1, \ldots, k$, the graph $G_\cap$ contains a copy $\overline{T_i}$ of the tree $T_i \in \mathcal{T}$, whose root is a vertex t_i, and contains two stars P_i and Q_i centered at p_i and q_i, respectively, whose number of leaves is determined as follows. For each vertex $u \in V_i$ such that an edge $(u, v) \in E$ exists connecting u to a vertex $v \in V_{i-1}$, the star P_i contains a leaf $u(P_i)$; also, for each

vertex $u \in V_i$ such that an edge $(u, v) \in E$ exists connecting u to a vertex $v \in V_{i+1}$, the star Q_i contains a leaf $u(Q_i)$. We also denote by $u(\overline{T_i})$ a leaf of $\overline{T_i}$ corresponding to a vertex $u \in V_i$.

The graph G_1^* contains $G_\cap$ plus the following edges. For $i = 1, \ldots, k$, consider each vertex $u \in V_i$. Suppose that i is even. Then, G_1^* has an edge connecting the leaf $u(\overline{T_i})$ of $\overline{T_i}$ corresponding to u with either the leaf $u(Q_i)$ of Q_i corresponding to u, if it exists, or with q_i, otherwise; also, for each edge in E connecting a vertex $u \in V_i$ with a vertex $v \in V_{i-1}$, the graph G_1^* has an edge connecting the leaf $u(P_i)$ of P_i corresponding to u with the leaf $v(Q_{i-1})$ of Q_{i-1} corresponding to v (these leaves exist by construction). Suppose that i is odd. Then, G_1^* has an edge between $u(\overline{T_i})$ and either $u(P_i)$, if it exists, or p_i, otherwise.

The graph G_2^* contains $G_\cap$ plus the following edges. For $i = 1, \ldots, k$, consider each vertex $u \in V_i$. Suppose that i is odd. Then, G_2^* has an edge connecting $u(\overline{T_i})$ with either the leaf $u(Q_i)$ of Q_i corresponding to u, if it exists, or with q_i, otherwise; also, for each edge in E connecting a vertex $u \in V_i$ with a vertex $v \in V_{i-1}$, the graph G_2^* has an edge $(u(P_i), v(Q_{i-1}))$. Suppose that i is even. Then, G_2^* has an edge between $u(\overline{T_i})$ and either $u(P_i)$, if it exists, or p_i, otherwise.

The graph $G_\cap$ is clearly connected. We prove that G_1^* and G_2^* are 2-connected, that is, removing any vertex v disconnects neither G_1^* nor G_2^*. If v is a leaf of $\overline{T_i}$, P_i, or Q_i, with $1 \leq i \leq k$, then removing v disconnects neither G_1^* nor G_2^*, since $G_\cap$ remains connected. If v is an internal node (the root) of $\overline{T_i}$, P_i, or Q_i, say of $\overline{T_i}$, with $1 \leq i \leq k$, then removing v disconnects $G_\cap$ into one component $\overline{T_i}(v)$ containing all the vertices of $\mathcal{C}$ (resp. all the vertices of $\mathcal{C}$, except for v) and into some subtrees $\overline{T_{i,j}}$ of $\overline{T_i}$ rooted the children of v; however, by construction, each leaf $u(\overline{T_i})$ of $\overline{T_{i,j}}$ is connected to $\overline{T_i}(v)$ via an edge of G_1^*, namely either $(u(\overline{T_i}), u(P_i))$, $(u(\overline{T_i}), p_i)$, $(u(\overline{T_i}), u(Q_i))$, or $(u(\overline{T_i}), q_i)$ (and similar for G_2^*), hence G_1^* (and G_2^*) is connected after the removal of v.

Observe that, if $n_\mathcal{T}$ denotes the total number of nodes in the trees in $\mathcal{T}$, then $\langle G_1^*, G_2^* \rangle$ contains at most $3n_\mathcal{T}$ vertices. Also, the number of edges of $\langle G_1^*, G_2^* \rangle$ is at most $|E| + 2n_\mathcal{T}$. Hence, the size of $\langle G_1^*, G_2^* \rangle$ is linear in the size of $(V, E, \gamma, \mathcal{T})$; also, it is easy to see that $\langle G_1^*, G_2^* \rangle$ can be constructed in linear time.

We prove that $\langle G_1^*, G_2^* \rangle$ admits a SEFE if and only if $(V, E, \gamma, \mathcal{T})$ is $\mathcal{T}$-level planar.

Suppose that $\langle G_1^*, G_2^* \rangle$ admits a SEFE $\langle \Gamma_1^*, \Gamma_2^* \rangle$. We show how to construct a drawing Γ of $(V, E, \gamma, \mathcal{T})$. For $1 \leq i \leq k$, let $\Theta(\overline{T_i})$ be the order in which the leaves of $\overline{T_i}$ appear in a pre-order traversal of $\overline{T_i}$ in $\langle \Gamma_1^*, \Gamma_2^* \rangle$; then, let the ordering $\mathcal{O}_i$ of the vertices of V_i along L_i be either $\Theta(\overline{T_i})$, if i is odd, or the reverse of $\Theta(\overline{T_i})$, if i is even.

We prove that Γ is $\mathcal{T}$-level planar. For each $i = 1, \ldots, k$, $\mathcal{O}_i$ is compatible with $T_i \in \mathcal{T}$, since the drawing of $\overline{T_i}$, that belongs to $G_\cap$, is planar in $\langle \Gamma_1^*, \Gamma_2^* \rangle$. Suppose,

for a contradiction, that two edges $(u, v), (w, z) \in E$ exist, with $u, w \in V_i$ and $v, z \in V_{i+1}$, that intersect in Γ. Hence, either u appears before w in $\mathcal{O}_i$ and v appears after z in $\mathcal{O}_{i+1}$, or vice versa. Since i and $i+1$ have different parity, either u appears before w in $\Theta(\overline{T_i})$ and v appears before z in $\Theta(\overline{T_{i+1}})$, or vice versa. We claim that, in both cases, this implies a crossing in $\langle \Gamma_1^*, \Gamma_2^* \rangle$ between paths $(q_i, u(Q_i), v(P_{i+1}), p_{i+1})$ and $(q_i, w(Q_i), z(P_{i+1}), p_{i+1})$ in $\langle G_1^*, G_2^* \rangle$. Since the edges of these two paths belong all to G_1^* or all to G_2^*, depending on whether i is even or odd, this yields a contradiction. We now prove the claim. The pre-order traversal $\Theta(Q_i)$ of Q_i (the pre-order traversal $\Theta(P_{i+1})$ of P_{i+1}) in $\langle \Gamma_1^*, \Gamma_2^* \rangle$ restricted to the leaves of Q_i (of P_{i+1}) is the reverse of $\Theta(\overline{T_i})$ (of $\Theta(\overline{T_{i+1}})$) restricted to the vertices of V_i (of V_{i+1}) corresponding to leaves of Q_i (of P_{i+1}). Namely, each leaf $x(Q_i)$ of Q_i ($y(P_{i+1})$ of P_{i+1}) is connected to the leaf $x(\overline{T_i})$ of $\overline{T_i}$ ($y(\overline{T_{i+1}})$ of $\overline{T_{i+1}}$) in the same graph, either G_1^* or G_2^*, by construction. Hence, the fact that u appears before (after) w in $\Theta(\overline{T_i})$ and v appears before (after) z in $\Theta(\overline{T_{i+1}})$ implies that u appears after (before) w in $\Theta(Q_i)$ and v appears after (before) z in $\Theta(P_{i+1})$. In both cases, this implies a crossing in $\langle \Gamma_1^*, \Gamma_2^* \rangle$ between the two paths.

Suppose that $(V, E, \gamma, \mathcal{T})$ admits a $\mathcal{T}$-level planar drawing Γ. We show how to construct a SEFE $\langle \Gamma_1^*, \Gamma_2^* \rangle$ of $\langle G_1^*, G_2^* \rangle$. For $1 \leq i \leq k$, let $\mathcal{O}_i$ be the order of the vertices of the level V_i along L_i in Γ. Since Γ is $\mathcal{T}$-level planar, there exists an embedding Γ_i of the tree $T_i \in \mathcal{T}$ that is compatible with $\mathcal{O}_i$. If i is odd (even), then assign to each internal vertex of $\overline{T_i}$ the same (resp. the opposite) rotation scheme as its corresponding vertex in Γ_i. Also, if i is odd, then assign to p_i (to q_i) the rotation scheme in G_1^* (resp. in G_2^*) such that the paths that connect p_i (resp. q_i) to the leaves of $\overline{T_i}$, either with an edge or passing through a leaf of P_i (resp. of Q_i), appear in the same clockwise order as the vertices of V_i appear in $\mathcal{O}_i$; if i is even, then assign to p_i (to q_i) the rotation scheme in G_2^* (resp. in G_1^*) such that the paths that connect p_i (resp. q_i) to the leaves of $\overline{T_i}$ appear in the same counterclockwise order as the vertices of V_i appear in $\mathcal{O}_i$. Finally, consider the embedding $\Gamma_{i,i+1}$ obtained by restricting Γ to the vertices and edges of the subgraph induced by the vertices of V_i and V_{i+1}. If i is odd (even), then assign to the leaves of Q_i and P_{i+1} in G_1^* (in G_2^*) the same rotation scheme as their corresponding vertices have in $\Gamma_{i,i+1}$. This completes the construction of $\langle \Gamma_1^*, \Gamma_2^* \rangle$.

We prove that $\langle \Gamma_1^*, \Gamma_2^* \rangle$ is a SEFE of $\langle G_1^*, G_2^* \rangle$. Since the rotation scheme of the internal vertices of each $\overline{T_i}$ are constructed starting from an embedding Γ_i of $T_i \in \mathcal{T}$ that is compatible with $\mathcal{O}_i$, the drawing of $\overline{T_i}$ is planar. Further, since the rotation schemes of p_i (of q_i) are also constructed starting from $\mathcal{O}_i$, there exists no crossing between two paths connecting t_i and p_i (t_i and q_i), one passing through a leaf $u(\overline{T_i})$ of $\overline{T_i}$ and, possibly, through a leaf $u(P_i)$ of P_i (through a leaf $u(Q_i)$ of Q_i), and the other passing through a leaf $v(\overline{T_i})$ of $\overline{T_i}$ and, possibly, through a leaf $v(P_i)$ of P_i (through

a leaf $v(Q_i)$ of Q_i). Finally, since the rotation schemes of the leaves of Q_i and P_{i+1} are constructed from the embedding $\Gamma_{i,i+1}$ obtained by restricting Γ to the vertices and edges of the subgraph induced by the vertices of V_i and V_{i+1}, there exist no two crossing edges between leaves of Q_i and P_{i+1}. □

We remark that a reduction from T-LEVEL PLANARITY to SEFE-2 was described by Schaefer in [Sch13]; however, the instances of SEFE-2 obtained from that reduction do not satisfy any conditions that make SEFE-2 known to be solvable in polynomial-time.

Theorem 7.3 *There exists a quadratic-time algorithm that decides whether a proper instance $(V, E, \gamma, \mathcal{T})$ of T-LEVEL PLANARITY is $\mathcal{T}$-level planar.*

Proof: The statement follows from Lemma 7.1 and from the existence of a quadratic-time algorithm [BR13] that decides whether an instance $\langle G_1, G_2 \rangle$ of SEFE-2 such that G_1 and G_2 are 2-connected and the common graph $G_\cap$ is connected admits a SEFE. □

CL-PLANARITY

In the following we show how to test in polynomial time the existence of a cl-planar drawing for a proper instance (V, E, γ, T) of CL-PLANARITY.

Let (V, E, γ, T) be a proper cl-graph, let μ be a cluster of T, and recall that V_μ denotes the subset of V composed of the leaves of the subtree of T rooted at μ. We say that (V, E, γ, T) is *μ-connected between two levels* V_i and V_{i+1} if there exist two vertices $u \in V_\mu \cap V_i$ and $v \in V_\mu \cap V_{i+1}$ such that edge $(u, v) \in E$. Also, let $\gamma_{\min}(\mu) = \min\{i \mid V_i \cap V_\mu \neq \emptyset\}$ and let $\gamma_{\max}(\mu) = \max\{i \mid V_i \cap V_\mu \neq \emptyset\}$. Then (V, E, γ, T) is *level-μ-connected* if it is μ-connected between levels V_i and V_{i+1} for each $i = \gamma_{\min}(\mu), \ldots, \gamma_{\max}(\mu) - 1$. Finally, (V, E, γ, T) is *level-connected* if it is μ-level-connected for each cluster $\mu \in T$.

Our strategy consists of first transforming a proper instance of CL-PLANARITY into an equivalent level-connected instance, and then transforming such a level-connected instance into an equivalent proper instance of T-LEVEL PLANARITY.

Lemma 7.2 *Let (V, E, γ, T) be a proper instance of* CL-PLANARITY. *An equivalent level-connected instance $(V^*, E^*, \gamma^*, T^*)$ of* CL-PLANARITY *whose size is quadratic in the size of (V, E, γ, T) can be constructed in quadratic time.*

Proof: The construction of $(V^*, E^*, \gamma^*, T^*)$ consists of two steps. See Fig. 7.4.

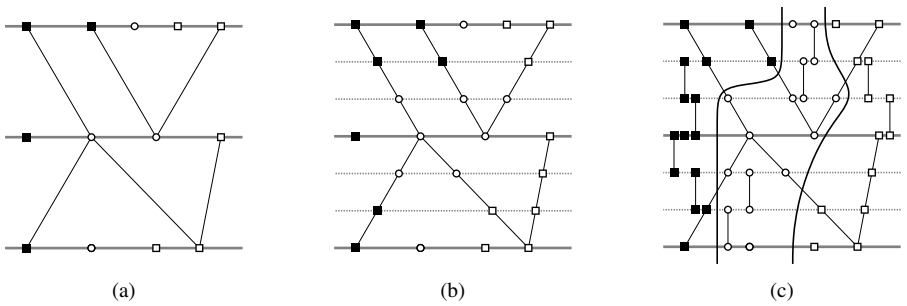


Figure 7.4: Illustration for the proof of Lemma 7.2. (a) An instance (V, E, γ, T) with flat hierarchy containing clusters $\mu_{\blacksquare}$, $\mu_{\square}$, and $\mu_{\circ}$. (b) Insertion of dummy vertices in (V, E, γ, T) to obtain (V', E', γ', T'). (c) The level-connected instance $(V^*, E^*, \gamma^*, T^*)$ obtained from (V', E', γ', T').

In the first step we turn (V, E, γ, T) into an equivalent instance (V', E', γ', T'). Initialize $V' = V$, $E' = E$, and $T' = T$. For each $i = 1, \ldots, k$ and for each vertex $u \in V_i$, set $\gamma'(u) = 3(i-1)+1$. Then, for each $i = 1, \ldots, k-1$, consider each edge $(u, v) \in E$ such that $\gamma(u) = i$ and $\gamma(v) = i+1$; add two vertices d_u and d_v to V', and replace (u, v) in E' with three edges (u, d_u), (d_u, d_v), and (d_v, v). Set $\gamma'(d_u) = 3(i-1)+2$ and $\gamma'(d_v) = 3i$. Finally, add d_u (d_v) to T' as a child of the parent of u (of v) in T'. The described transformation can be easily performed in linear time. Moreover, the size of (V', E', γ', T') is linear in the size of (V, E, γ, T).

We prove that (V', E', γ', T') is equivalent to (V, E, γ, T).

Suppose that (V, E, γ, T) admits a cl-planar drawing Γ; a cl-planar drawing Γ' of (V', E', γ', T') is constructed as follows. Initialize $\Gamma' = \Gamma$. Scale Γ' up by a factor of 3 and vertically translate it so that the vertices in V'_1 lie on the line $y = 1$. After the two affine transformations have been applied (i) Γ' has no crossing, (ii) every edge is a y-monotone curve, (iii) for $i = 1, \ldots, k$, the vertices in $V_i = V'_{3(i-1)+1}$ are placed on the line $y = 3(i-1)+1$, that we denote by $L'_{3(i-1)+1}$, and (iv) the order in which the vertices in $V_i = V'_{3(i-1)+1}$ appear along $L'_{3(i-1)+1}$ is the same as the order in which they appear along L_i. For each $i = 1, \ldots, k-1$, consider each edge $(u, v) \in E$ such that $\gamma(u) = i$ and $\gamma(v) = i+1$. Place the vertices d_u and d_v in Γ' on the two points of the curve representing (u, v) having y-coordinate equal to $3(i-1)+2$ and $3i$, respectively. Then the curves representing any two edges in E' are part of the curves representing any two edges in E. Hence Γ' is a cl-planar drawing of (V', E', γ', T').

Suppose that (V', E', γ', T') admits a cl-planar drawing Γ'; a cl-planar drawing Γ

of (V, E, γ, T) is constructed as follows. Initialize $\Gamma = \Gamma'$. For $i = 1, \ldots, k-1$, consider each path (u, d_u, d_v, v) such that $\gamma'(u) = 3(i-1)+1$ and $\gamma'(v) = 3i+1$; remove d_u, d_v, and their incident edges in E' from Γ; draw the edge $(u, v) \in E$ in Γ as the composition of the curves representing the edges (u, d_u), (d_u, d_v), and (d_v, v) in Γ'. Scale Γ down by a factor of 3 and vertically translate it so that the vertices of V_1 lie on the line $y = 1$. After the two affine transformations have been applied (i) Γ has no crossing, (ii) every edge is a y-monotone curve, (iii) for $i = 1, \ldots, k$, the vertices of V_i are placed on the line $y = i$, and (iv) the order in which the vertices in $V_i = V'_{3(i-1)+1}$ appear along L_i is the same as the order in which they appear along $L'_{3(i-1)+1}$. Since Γ' is cl-planar, this implies that Γ is cl-planar, as well.

The goal of this transformation was to obtain an instance (V', E', γ', T') such that, if there exists a vertex $u \in V'_j$, with $1 \leq j \leq 3(k-1)+1$, that is adjacent to two vertices $v, w \in V'_h$, with $h = j \pm 1$, then u, v, and w have the same parent node $\mu \in T'$; hence, (V', E', γ', T') is μ-connected between levels V'_j and V'_h.

In the second step we transform (V', E', γ', T') into an equivalent level-connected instance $(V^*, E^*, \gamma^*, T^*)$. Initialize $(V^*, E^*, \gamma^*, T^*) = (V', E', \gamma', T')$. Consider each cluster $\mu \in T'$ according to a bottom-up visit of T'. If there exists a level V'_i, with $\gamma'_{\min}(\mu) \leq i < \gamma'_{\max}(\mu)$, such that no edge in E' connects a vertex $u \in V'_i \cap V'_\mu$ with a vertex $v \in V'_{i+1} \cap V'_\mu$, then add two vertices u^* and v^* to V^*, add an edge (u^*, v^*) to E^*, set $\gamma^*(u^*) = i$ and $\gamma^*(v^*) = i+1$, and add u^* and v^* to T^* as children of μ.

Observe that, for each cluster $\mu \in T'$ and for each level $1 \leq i \leq 3k-2$, at most two dummy vertices are added to $(V^*, E^*, \gamma^*, T^*)$. This implies that the size of $(V^*, E^*, \gamma^*, T^*)$ is quadratic in the size of (V', E', γ', T') and hence in the size of (V, E, γ, T). Also, the whole construction can be performed in quadratic time.

It remains to prove that $(V^*, E^*, \gamma^*, T^*)$ is equivalent to (V', E', γ', T').

Suppose that $(V^*, E^*, \gamma^*, T^*)$ admits a cl-planar drawing Γ^*; a cl-planar drawing Γ' of (V', E', γ', T') can be constructed as follows. Initialize $\Gamma' = \Gamma^*$ and remove from V', E', and Γ' all the vertices and edges added when constructing Γ^*. Since all the other vertices of V' and edges of E' have the same representation in Γ' and in Γ^*, and since Γ^* is cl-planar, it follows that Γ' is cl-planar, as well.

Suppose that (V', E', γ', T') admits a cl-planar drawing Γ'; a cl-planar drawing Γ^* of $(V^*, E^*, \gamma^*, T^*)$ can be constructed as follows. Initialize $\Gamma^* = \Gamma'$. Consider a level V'_i, with $1 \leq i \leq 3(k-1)$, such that the vertices $u^*, v^* \in \mu$ with $\gamma'(u^*) = i$ and $\gamma'(v^*) = i+1$, for some cluster $\mu \in T$, have been added to $(V^*, E^*, \gamma^*, T^*)$. By construction, (V', E', γ', T') is not μ-connected between the levels V'_i and V'_{i+1}. As observed before, this implies that no vertex $u \in V'_i \cap V'_\mu$ exists that is connected to two vertices $v, w \in V'_{i+1}$, and no vertex $u \in V'_{i+1} \cap V'_\mu$ exists that is connected to

two vertices $v, w \in V'_i$. Hence, u^*, v^*, and the edge (u^*, v^*) connecting them can be drawn in Γ^* entirely inside the region representing μ in such a way that u^* and v^* lie along the lines L'_i and L'_{i+1} and there exists no crossing between (u^*, v^*) and any other edge.

This concludes the proof of the lemma. □

Lemma 7.3 *Let (V, E, γ, T) be a level-connected instance of* CL-PLANARITY. *An equivalent proper instance $(V, E, \gamma, \mathcal{T})$ of* T-LEVEL PLANARITY *whose size is linear in the size of (V, E, γ, T) can be constructed in quadratic time.*

Proof: We construct $(V, E, \gamma, \mathcal{T})$ from (V, E, γ, T) as follows. Initialize $\mathcal{T} = \emptyset$. For $i = 1, \ldots, k$, add to $\mathcal{T}$ a tree T_i that is the subtree of the cluster hierarchy T whose leaves are all and only the vertices of level V_i.

We prove that $(V, E, \gamma, \mathcal{T})$ is $\mathcal{T}$-level planar if and only if (V, E, γ, T) is cl-planar.

Suppose that $(V, E, \gamma, \mathcal{T})$ admits a $\mathcal{T}$-level planar drawing Γ^*; we show how to construct a cl-planar drawing Γ of (V, E, γ, T). Initialize $\Gamma = \Gamma^*$. Consider each level V_i, with $i = 1, \ldots, k$. By construction, for each cluster $\mu \in T$ such that there exists a vertex $v \in V_i \cap V_\mu$, there exists an internal node of the tree $T_i \in \mathcal{T}$ whose leaves are all and only the vertices of $V_i \cap V_\mu$. Since Γ^* is $\mathcal{T}$-level planar, such vertices appear consecutively along L_i. Hence, in order to prove that Γ is a cl-planar drawing, it suffices to prove that there exist no four vertices u, v, w, z such that (i) $u, v \in V_i$ and $w, z \in V_j$, with $1 \leq i < j \leq k$; (ii) $u, w \in V_\mu$ and $v, z \in V_\nu$, with $\mu \neq \nu$; and (iii) u appears before v on L_i and w appears after z on L_j, or vice versa. Suppose, for a contradiction, that such four vertices exist. We can assume $j = i \pm 1$ without loss of generality, as (V, E, γ, T) is level-connected. Assume that u appears before v along L_i and w appears after z along L_j, the other case being symmetric. Since Γ^* is $\mathcal{T}$-level planar, all the vertices of V_μ appear before all the vertices of V_ν along L_i and all the vertices of V_μ appear after all the vertices of V_ν along L_j. Also, since (V, E, γ, T) is level-connected, there exists at least an edge (a, b) such that $a \in V_i \cap V_\mu$ and $b \in V_j \cap V_\mu$, and an edge (c, d) such that $c \in V_i \cap V_\nu$ and $d \in V_j \cap V_\nu$. However, under the above conditions, these two edges intersect in Γ and in Γ^*, hence contradicting the hypothesis that Γ^* is $\mathcal{T}$-level planar.

Suppose that (V, E, γ, T) admits a cl-planar drawing Γ; we show how to construct a $\mathcal{T}$-level planar drawing Γ^* of $(V, E, \gamma, \mathcal{T})$. Initialize $\Gamma^* = \Gamma$. Consider each level V_i, with $i = 1, \ldots, k$. By construction, for each internal node w of the tree $T_i \in \mathcal{T}$, there exists a cluster $\mu \in T$ such that the vertices of $V_i \cap V_\mu$ are all and only the leaves of the subtree of T_i rooted at w. Since Γ is cl-planar, such vertices appear consecutively along L_i. Hence, Γ^* is $\mathcal{T}$-level planar.

The construction of $(V, E, \gamma, \mathcal{T})$ can be easily performed in quadratic time by visiting T a number of times equal to the number of levels of (V, E, γ, T).

The size of $(V, E, \gamma, \mathcal{T})$ might be quadratic in the size of (V, E, γ, T). However, $(V, E, \gamma, \mathcal{T})$ can be modified so that its size is linear in the size of (V, E, γ, T) as follows. Consider the tree T_i, for each $i = 1, \ldots, k$. Replace each path $(\nu_1, \ldots, \nu_\ell)$ in T_i such that ν_j is the parent of ν_{j+1}, for $i = 1, \ldots, \ell-1$, and such that ν_j has no children other than ν_{j+1}, for $i = 2, \ldots, \ell - 1$, with a single edge (ν_1, ν_ℓ). After this transformation, each internal node of T_i has at least two children, hence the size of T_i is linear in the size of V_i. Since the vertex sets of $(V, E, \gamma, \mathcal{T})$ and (V, E, γ, T) coincide, it follows that the size of $(V, E, \gamma, \mathcal{T})$ is linear in the size of (V, E, γ, T). Finally, the described transformation does not alter the T-LEVEL PLANARITY of $(V, E, \gamma, \mathcal{T})$. This concludes the proof of the lemma. □

We get the following.

Theorem 7.4 *There exists a quartic-time algorithm that decides whether a proper instance* (V, E, γ, T) *of* CLUSTERED-LEVEL PLANARITY *is cl-planar.*

Proof: By Lemma 7.2, a level-connected instance (V', E', γ', T') of CL-PLANARITY can be constructed that is cl-planar if and only if (V, E, γ, T) is cl-planar. The construction can be accomplished in quadratic time in the size of (V, E, γ, T); moreover, the size of (V', E', γ', T') is quadratic in the size of (V, E, γ, T).

By Lemma 7.3, a proper instance $(V', E', \gamma', \mathcal{T}')$ of T-LEVEL PLANARITY can be constructed that is $\mathcal{T}$-level planar if and only if (V', E', γ', T') is cl-planar. The construction can be accomplished in linear time in the size of (V', E', γ', T'), and hence in quadratic time in the size of (V, E, γ, T); moreover, the size of $(V', E', \gamma', \mathcal{T}')$ is linear in the size of (V', E', γ', T'), and hence quadratic in the size of (V, E, γ, T).

Finally, by Theorem 7.3, it is possible to test whether $(V', E', \gamma', \mathcal{T}')$ is $\mathcal{T}$-level planar in quadratic time in the size of $(V', E', \gamma', \mathcal{T}')$, and hence in quartic time in the size of (V, E, γ, T). □

7.4 Open Problems

Several problems are opened by this research:

1. The algorithms for testing level planarity [JLM98] and for testing cl-planarity for level-connected proper hierarchies [FB04] both have linear-time complexity. Although our algorithms solve more general problems than the ones above, they are less efficient. This leaves room for future research aiming at improving our complexity bounds.

2. Our NP-hardness result on the complexity of CL-PLANARITY exploits a cluster hierarchy whose depth is linear in the number of vertices of the underlying graph. Does the NP-hardness hold if the cluster hierarchy is flat?

3. The NP-hardness of CL-PLANARITY is, to the best of our knowledge, the first hardness result for a variation of the clustered planarity problem in which none of the c-planarity constraints is dropped. Is it possible to use similar techniques to tackle the problem of determining the complexity of CLUSTERED PLANARITY?

Part IV

Simultaneous Embedding with Fixed Edges

Chapter 8

Advancements on SEFE and Partitioned Book Embedding Problems

In this chapter[1] we investigate the complexity of some combinatorial problems related to the SIMULTANEOUS EMBEDDING WITH FIXED EDGES (SEFE) and the PARTITIONED T-COHERENT k-PAGE BOOK EMBEDDING (PTBE-k) problems, which are known to be equivalent under certain conditions.

The SEFE problem is NP-complete for $k \geq 3$ even if the intersection graph is the same for each pair of graphs (*sunflower intersection*). We prove that this is true even when the intersection graph is a tree and all the input graphs are biconnected. This result implies the NP-completeness of PTBE-k for $k \geq 3$. However, we prove stronger results on this problem, namely that PTBE-k remains NP-complete for $k \geq 3$ even if (i) two of the input graphs $G_i = T \cup E_i$ are biconnected and T is a caterpillar or if (ii) T is a star. This latter setting is also known in the literature as PARTITIONED k-PAGE BOOK EMBEDDING. On the positive side, we provide a linear-time algorithm for PTBE-k when all but one of the edge-sets induce connected graphs.

Finally, we prove that the problem of maximizing the number of edges that are drawn the same in a SEFE of two graphs (*optimization of* SEFE) is NP-complete, even in several restricted settings.

[1] The contents of this chapter are a joint work with Patrizio Angelini and Daniel Neuwirth, appeared partially in [ADN14] and in a journal [ADN15].

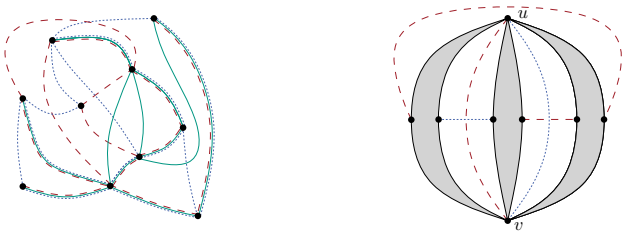
8.1 Introduction

The possibility of drawing together a set of graphs gives the opportunity to represent at the same time a set of different binary relationships among the same objects or a single relationship evolving over time, hence making this topic a fundamental tool in Information Visualization [EKLN05]. Motivated by such applications and by their theoretical appeal, simultaneous graph embeddings received wide research attention in the last few years. For an up-to-date survey, see [BKR13b].

Recently, a new major milestone to assert the importance of SEFE has been provided by Schaefer [Sch13], who discussed its relationships with some other famous problems in Graph Drawing, proving that SEFE generalizes several of them. In particular, he showed a polynomial-time reduction to SEFE with $k = 2$ from the C-PLANARITY problem, whose computational complexity is still one of the most important open questions in Graph Drawing. Recently, the reduction in the opposite direction has been proved [AD14], but only for instances of SEFE of two graphs in which the intersection graph is connected. Chapter 9 is mostly devoted to the description of such a new reduction. We remark that this "connected" version of SEFE is equivalent to problem PTBE-k for $k = 2$ [ADF$^+$12].

We refer the reader to Section 3.2 for an exhaustive survey of the state of the art of the computational complexity of the SEFE and of the PTBE-k problems before the contributions of this chapter (see also [ADN15] and [ADN14]).

In Chapter 11 of the Handbook of Graph Drawing and Visualization [BKR13b], the SEFE problem with *sunflower intersection* (SUNFLOWER SEFE) is reported as an open question (Open Problem 7). As described in Section 3.2, in this setting the intersection graph $G_\cap$ is such that, if an edge belongs to $G_\cap$, then it belongs to all the input graphs. See Fig. 8.1(a) for an example. Note that every instance of SEFE with $k = 2$ obviously has sunflower intersection. We remark that the same technique used in [ADF$^+$12] to prove that SEFE-2 of two graphs with connected intersection is equivalent to PTBE-2 can be applied to prove that SUNFLOWER SEFE of k graphs with connected intersection is equivalent to PTBE-k. Haeupler *et al.* [HJL13] conjectured that SUNFLOWER SEFE is polynomial-time solvable. However, Schaefer [Sch13] recently proved that this problem is NP-complete for $k \geq 3$ by providing a reduction from PTBE-k. Observe that, this reduction produces instances of SUNFLOWER SEFE in which the intersection graph is a spanning forest composed of an unbounded number of star graphs [Sch13].



(a) (b)

Figure 8.1: (a) A SUNFLOWER SEFE of three planar graphs. (b) A MAX SEFE of two graphs. Note that edge (u, v) is represented as a different curve in the two drawings.

Our Results

In this chapter, we prove that SUNFLOWER SEFE is NP-complete for $k \geq 3$ even if $G_\cap$ is a single spanning tree and all the input graphs are biconnected. We remark that having higher connectivity, both on the input graphs and on their intersection, is often a key factor to obtain polynomial-time solutions for this problem [ADF$^+$12, HJL13, BKR13a, BR13].

Given the equivalence between the connected version of SUNFLOWER SEFE and PTBE-k [ADF$^+$12], our result implies the NP-completeness of PTBE-k for $k \geq 3$; however, the biconnectivity of the graphs in SUNFLOWER SEFE is not maintained in the reduction to PTBE-k, that is, instances $\langle T, E_1, \ldots, E_k \rangle$ produced by the reduction are such that graphs $G_i = T \cup E_i$ are possibly not biconnected. In this direction, we investigate the complexity of PTBE-k under stronger assumptions on the connectivity of the input graphs and show that it remains NP-complete for $k \geq 3$ even if two of the input graphs G_i are biconnected. Further, we prove NP-completeness for this problem when T is a star; this setting, in which the tree T basically does not impose any constraint on the ordering of the vertices, is also known as PARTITIONED k-PAGE BOOK EMBEDDING (PBE-k). Since PBE-k with $k = 2$ can be solved in linear time [HN14], this result is tight.

From the algorithmic point of view, we prove that PTBE-k with $k \geq 2$ can be solved in linear time if $k - 1$ of the input edge-sets E_i induce connected graphs (note that, this is a stronger condition than graph G_i being biconnected), hence improving on a result by Hoske [Hos12], that requires all the k input edge-sets to have this property. Of course, relaxing this constraint on one of the k input edge-sets becomes

Problem	$G_\cap$	k	Bico	$\mathcal{T}$-Bico	Complexity
SUNFLOWER	tree	$k \geq 3$	k	–	NPC (Th.8.1)
PBE-k	star	$k \geq 3$	–	–	NPC (Th. 8.4)
PBE-2	star	$k = 2$	–	–	$O(n)$ ([HN14])
PTBE-3	caterpillar	$k \geq 3$	2	–	NPC (Th. 8.2)
PTBE-k	tree	$k \geq 2$	$k-1$	$k-1$	$O(n)$ (Th. 8.5)
PTBE-2	tree	$k = 2$	2	–	$O(n^2)$ ([BR13])
	binary tree	$k = 2$	–	–	$O(n^2)$ ([Hos12])
	tree	$k = 2$	1	–	OPEN (Th. 8.6)

Table 8.1: Complexity status for SUNFLOWER SEFE, PTBE-k, and PBE-k. Columns with labels Bico and $\mathcal{T}$-Bico report how many of the input graphs are biconnected and $\mathcal{T}$-biconnected, respectively.

more relevant for small values of k; in particular, it contributes to extend the class of instances that can be solved in polynomial time also for $k = 2$.

An updated summary of the results on SUNFLOWER SEFE and on PTBE-k is presented in Table 8.1.

For the setting $k = 2$ we also prove that, given any instance of PTBE-k (and hence of SEFE in which $G_\cap$ is connected), it is possible to construct an equivalent instance of the same problem in which one of the input graphs, say G_1, is biconnected and series-parallel. This implies that it would be sufficient to find a polynomial-time algorithm for this seemingly restricted case in order to have a polynomial-time algorithm for the whole problem.

Finally, still in the setting $k = 2$, we study the optimization version of SEFE, that we call MAX SEFE, which is cited as an open question by Haeupler *et al.* [HJL13] and in Chapter 11 (Open Problem 9) of the Handbook of Graph Drawing and Visualization [BKR13b]. In this problem, one asks for drawings of G_1 and G_2 such that as many edges of $G_\cap$ as possible are drawn the same. See Fig. 8.1(b) for an example. We prove that MAX SEFE is NP-complete, even under some strong constraints. Namely, the problem is NP-complete if G_1 and G_2 are triconnected, and $G_\cap$ is composed of a cubic triconnected component plus a set of isolated vertices. This implies that the problem is computationally hard both in the fixed and in the variable embedding case. In the latter case, however, we can prove that MAX SEFE is NP-complete even if $G_\cap$ has degree at most 2. Observe that any of these constraints would be sufficient to obtain polynomial-time algorithms for the original decision version of SEFE with $k = 2$.

The chapter is structured as follows. In Sect. 8.2 we deal with the sunflower intersection scenario; in Sect. 8.3 we focus on the PTBE-k problem; while in Sect. 8.4 we study the MAX SEFE problem. Finally, in Sect. 8.5 we give concluding remarks and discuss some open problems.

8.2 Sunflower SEFE

In this section we study the SUNFLOWER SEFE problem, that is the restriction of SEFE to instances in which the intersection graph $G_\cap$ is the same for each pair of graphs, that is, $G_\cap = G_i \cap G_j$ for each $1 \leq i < j \leq k$. We prove that SUNFLOWER SEFE is NP-complete with $k \geq 3$ even if $G_\cap$ is a spanning tree and all the input graphs are biconnected.

The proof is based on a polynomial-time reduction from the NP-complete [Opa79] problem BETWEENNESS, that takes as input a finite set $A = \{1, \ldots, n\}$ of n objects and a set $C \subseteq A \times A \times A$ of m ordered triples of distinct elements of A, and asks whether a linear ordering $\mathcal{O}$ of the elements of A exists such that for each triple $\langle \alpha, \beta, \gamma \rangle \in C$, we have either $\mathcal{O} =< \ldots, \alpha, \ldots, \beta, \ldots, \gamma, \ldots >$ or $\mathcal{O} =< \ldots, \gamma, \ldots, \beta, \ldots, \alpha, \ldots >$.

In order to simplify the proof, we first give in Lemma 8.1 an NP-completeness proof for a less restricted setting of SUNFLOWER SEFE and then describe how the produced instances can be modified in order to obtain equivalent instances with the desired properties.

Lemma 8.1 SUNFLOWER SEFE *with $k = 3$ is NP-complete even if two of the input graphs are biconnected and the intersection graph $G_\cap$ is a spanning pseudo-tree.*

Proof: The membership in NP of SUNFLOWER SEFE descends from that of SEFE, which has been proved in [GJP+06] by a reduction to the *Weak Realizability* problem [Kra98, KLN91].

The NP-hardness is proved by means of a polynomial-time reduction from problem BETWEENNESS. Given an instance $\langle A, C \rangle$ of BETWEENNESS, we construct an instance $\langle G_1, G_2, G_3 \rangle$ of SUNFLOWER SEFE that admits a SEFE if and only if $\langle A, C \rangle$ is a positive instance of BETWEENNESS, as follows.

Refer to Fig. 8.2 for an illustration of the construction of $G_\cap$, G_1, G_2, and G_3.

Graph $G_\cap$ contains a cycle $\mathcal{C} = u_1, v_1, u_2, v_2, \ldots, u_m, v_m, w_m, \ldots, w_1$ of $3m$ vertices. Also, for each $i = 1, \ldots, m$, $G_\cap$ contains a star S_i with n leaves centered at u_i and a star T_i with n leaves centered at v_i. For each $i = 1, \ldots, m$, the leaves of S_i are labeled x_i^j and the leaves of T_i are labeled y_i^j, for $j = 1, \ldots, n$. Graph G_1 contains all the edges of $G_\cap$ plus a set of edges (y_i^j, x_{i+1}^j), for $i = 1, \ldots, m$

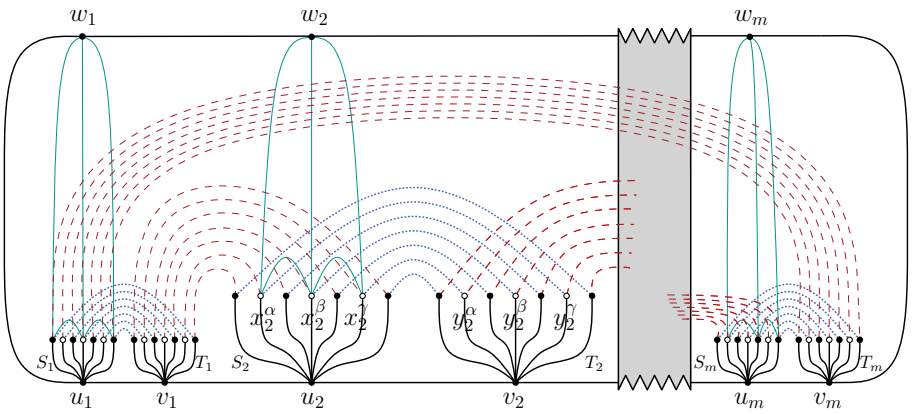


Figure 8.2: Illustration of the composition of $G_\cap$, G_1, G_2, and G_3 in Lemma 8.1, focused on triple $t_2 = \langle \alpha, \beta, \gamma \rangle$ of C. The three leaves of each star S_i and T_i corresponding to triple the elements of t_2 are represented as white circles.

and $j = 1, \ldots, n$. Here and in the following, $i + 1$ is computed modulo m. Graph G_2 contains all the edges of $G_\cap$ plus a set of edges (x_i^j, y_i^j), for $i = 1, \ldots, m$ and $j = 1, \ldots, n$. Graph G_3 contains all the edges of $G_\cap$ plus a set of edges defined as follows. For each $i = 1, \ldots, m$, consider the i-th triple $t_i = \langle \alpha, \beta, \gamma \rangle$ of C, and the corresponding vertices x_i^α, x_i^β, and x_i^γ of S_i; graph G_3 contains edges (w_i, x_i^α), (w_i, x_i^β), (w_i, x_i^γ), (x_i^α, x_i^β), and (x_i^β, x_i^γ).

First note that, by construction, $\langle G_1, G_2, G_3 \rangle$ is an instance of SUNFLOWER SEFE and graph $G_\cap$ is a spanning pseudo-tree. Also, one can easily verify that G_1 and G_2 are biconnected.

We claim that, in any SEFE of $\langle G_1, G_2, G_3 \rangle$, the following two properties hold:

Property 1 for each $i = 1, \ldots, m$, the ordering of the edges of S_i around u_i is the same as the ordering of the edges of S_{i+1} around u_{i+1}, where the vertices of S_i and S_{i+1} are identified based on j.

Property 2 for each $i = 1, \ldots, m$, edges (u_i, x_i^α), (u_i, x_i^β), and (u_i, x_i^γ), corresponding to triple $t_i = \langle \alpha, \beta, \gamma \rangle$ in C, appear either in this order or in the reverse order around u_i.

To prove Property 1, note that the ordering of the edges of S_i around u_i is reversed with respect to the ordering of the edges of T_i around v_i, due to the presence of the

edges of G_2 between the leaves of S_i and the leaves of T_i. Also, the ordering of the edges of T_i around v_i is reversed with respect to the ordering of the edges of S_{i+1} around u_{i+1}, due to the presence of the edges of G_1 between the leaves of T_i and the leaves of S_{i+1}. The proof of Property 2 descends from the fact that the subgraph of G_3 induced by vertices w_i, u_i, x_i^α, x_i^β, and x_i^γ is such that adding edge (u_i, w_i) would make it triconnected, and hence it admits exactly two planar embeddings, which differ by a flip.

In the following we prove that $\langle G_1, G_2, G_3\rangle$ is a positive instance if and only if $\langle A, C\rangle$ is a positive instance of BETWEENNESS.

Suppose that $\langle G_1, G_2, G_3\rangle$ is a positive instance, that is, there exists a SEFE $\langle \Gamma_1, \Gamma_2, \Gamma_3\rangle$.

We construct a linear ordering $\mathcal{O}$ of the elements of A from the ordering of the leaves of S_1 in $\langle \Gamma_1, \Gamma_2, \Gamma_3\rangle$. Starting from the edge of S_1 clockwise following (u_1, w_1) around u_1, consider the leaves $x_1^{d_1}, \ldots, x_1^{d_n}$ of S_1 as they appear in clockwise order around u_1, with $d_j \in A$ for each $1 \leq j \leq n$. Then, set $\mathcal{O} = d_1, \ldots, d_n$.

We prove that $\mathcal{O}$ is a solution of $\langle A, C\rangle$. By Property 1, the clockwise ordering of the leaves of S_i is the same for every $i = 1, \ldots, m$. Also, by Property 2, for each triple $t_i = \langle \alpha, \beta, \gamma\rangle$, edges (u_i, x_i^α), (u_i, x_i^β), and (u_i, x_i^γ) appear around u_i either in this order or in the reverse one. Hence, α, β, and γ appear in $\mathcal{O}$ either in this order or in the reverse one, which implies that $\mathcal{O}$ is a solution of $\langle A, C\rangle$.

Suppose that $\langle A, C\rangle$ is a positive instance, that is, there exists an ordering $\mathcal{O} = d_1, \ldots, d_n$ of the elements of A in which for each triple t_i of C, the three elements of t_i appear in one of their two admissible orderings.

We construct an embedding for G_1,G_2, and G_3. For each $i = 1, \ldots, m$, the rotation schemes of u_i and v_i in all the three graphs are constructed as follows. The rotation scheme of u_i is (u_i, v_{i-1}), $(u_i, x_i^{d_1})$, $\ldots$, $(u_i, x_i^{d_n})$, (u_i, v_i), and the rotation scheme of v_i is (v_i, u_i), $(v_i, y_i^{d_n})$, $\ldots$, $(v_i, y_i^{d_1})$, (v_i, u_{i+1}), where $v_0 = w_1$ and $u_{m+1} = w_m$. Since all the vertices of G_1 and of G_2 different from u_i and v_i ($i = 1, \ldots, m$) have degree 2, the embeddings Γ_1 and Γ_2 of G_1 and G_2 are completely specified. To complete the embedding Γ_3 of G_3, we have to specify the rotation scheme of w_i and of the three leaves of S_i adjacent to w_i, for each $i = 1, \ldots, m$. Consider triple $t_i = \langle \alpha, \beta, \gamma\rangle$ of C. If α, β, and γ appear in this order in $\mathcal{O}$ (see Fig. 8.2), then the rotation scheme of w_i is (w_i, w_{i+1}), (w_i, x_i^γ), (w_i, x_i^β), (w_i, x_i^α), (w_i, w_{i-1}); the rotation scheme of x_i^α is (x_i^α, w_i), (x_i^α, x_i^β), (x_i^α, u_i); the rotation scheme of x_i^β is (x_i^β, w_i), (x_i^β, x_i^γ), (x_i^β, u_i), (x_i^β, x_i^α); and the rotation scheme of x_i^γ is (x_i^γ, w_i), (x_i^γ, u_i), (x_i^γ, x_i^β), where $w_0 = u_1$ and $w_{m+1} = v_m$. If α, β, and γ appear in the reverse order in $\mathcal{O}$, then the rotation scheme of w_i is (w_i, w_{i+1}), (w_i, x_i^α), (w_i, x_i^β), (w_i, x_i^γ), (w_i, w_{i-1}); the rotation scheme of x_i^α is (x_i^α, w_i), (x_i^α, u_i), (x_i^α, x_i^β); the ro-

tation scheme of x_i^β is (x_i^β, w_i), (x_i^β, x_i^α), (x_i^β, u_i), (x_i^β, x_i^γ); and the rotation scheme of x_i^γ is (x_i^γ, w_i), (x_i^γ, x_i^β), (x_i^γ, u_i), where $w_0 = u_1$ and $w_{m+1} = v_m$.

In order to prove that $\langle \Gamma_1, \Gamma_2, \Gamma_3 \rangle$ is a SEFE, we first observe that the embeddings of $G_\cap$ obtained by restricting Γ_1, Γ_2, and Γ_3 to the edges of $G_\cap$, respectively, coincide by construction.

In order to prove the planarity of Γ_1 (of Γ_2), observe that v_i and u_{i+1} (v_i and u_i), for each $i = 1, \ldots, m$, are the poles of a parallel subgraph composed of m paths. Since the order of the edges around v_i in Γ_1 (in Γ_2) is the reverse of the order of the edges around u_{i+1} (around u_i), these paths can be drawn without intersections.

The planarity of Γ_3 is due to the fact that, by construction, for each $i = 1, \ldots, m$, the subgraph induced by w_i, u_i, x_i^α, x_i^β, and x_i^γ is planar in Γ_3.

This concludes the proof of the lemma. □

We are now ready to prove the main result of the section, by showing how to extend the reduction of Lemma 8.1 to obtain instances with $k \geq 3$ in which all graphs are biconnected and $G_\cap$ is a tree.

Theorem 8.1 SUNFLOWER SEFE *is NP-complete for $k \geq 3$ even if all the input graphs are biconnected and the intersection graph is a spanning tree.*

Proof: The membership in NP has been proved in [GJP+06].

The NP-hardness is proved by means of a polynomial-time reduction from problem BETWEENNESS. Given an instance $\langle A, C \rangle$ of BETWEENNESS, we first construct an instance $\langle G_1^*, G_2^*, G_3^* \rangle$ of SUNFLOWER SEFE that admits a SEFE if and only if $\langle A, C \rangle$ is a positive instance of BETWEENNESS by applying the reduction shown in Lemma 8.1. We show how to modify $\langle G_1^*, G_2^*, G_3^* \rangle$ to obtain an equivalent instance $\langle G_1, G_2, G_3 \rangle$ with the required properties.

Refer to Fig. 8.3 for an illustration of the construction of $G_\cap$, G_1, G_2, and G_3.

Graph $G_\cap$ is initialized to $G_\cap^*$. For $i = 1, \ldots, m$, subdivide edge (w_u, w_{i+1}) (where $w_{m+1} = v_m$) with two vertices s_i and t_i, add a star with 3 leaves α_i, β_i, and γ_i with center c_i, and add an edge connecting w_i to c_i. Graph G_1 contains all the edges of $G_\cap$ plus a set of edges defined as follows. As in $\langle G_1^*, G_2^*, G_3^* \rangle$, for $i = 1, \ldots, m$, graph G_1 contains edges (y_i^j, x_{i+1}^j), with $j = 1, \ldots, n$, connecting the leaves of T_i to the leaves of S_{i+1}. Additionally, for $i = 1, \ldots, m$, G_1 contains edges (w_i, α_i),(α_i, β_i),(β_i, γ_i), (γ_i, w_i), and (β_i, s_i). Here and in the following, $i+1$ is computed modulo m. Graph G_2 contains all the edges of $G_\cap$ plus a set of edges defined as follows. As in $\langle G_1^*, G_2^*, G_3^* \rangle$, for $i = 1, \ldots, m$, graph G_2 contains edges (x_i^j, y_i^j), with $j = 1, \ldots, n$. Additionally, for $i = 1, \ldots, m$, G_2 contains edges (α_i, t_i), (β_i, t_i), and (γ_i, t_i). Graph G_3 contains all the edges of $G_\cap$ plus a set of edges defined as follows. For each $i = 1, \ldots, m$, consider the i-th triple $t_i = \langle \alpha, \beta, \gamma \rangle$

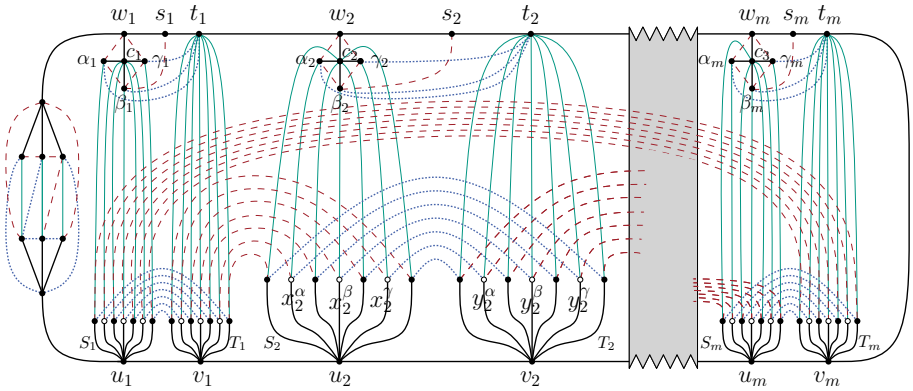


Figure 8.3: Illustration of the composition of $G_\cap$, G_1, G_2, and G_3 in Theorem 8.1, focused on triple $t_2 = \langle \alpha, \beta, \gamma \rangle$ of C.

of C, and the corresponding vertices x_i^α, x_i^β, and x_i^γ of S_i; graph G_3 contains edges (α_i, x_i^α), (β_i, x_i^β), (γ_i, x_i^γ), and edges (x_i^j, c_i), for every $j \notin \{\alpha, \beta, \gamma\}$. Also, for $i = 1, \ldots, m$, graph G_3 contains edges (y_i^j, t_i), with $j = 1, \ldots, n$.

Observe that, graph $G_\cap$ is a pseudo-tree and graphs G_1, G_2, and G_3 are biconnected. We first prove that the constructed instance $\langle G_1, G_2, G_3 \rangle$ of SUNFLOWER SEFE is equivalent to instance $\langle A, C \rangle$ of BETWEENNESS. Then, we show how to modify $\langle G_1, G_2, G_3 \rangle$ in such a way that $G_\cap$ is a tree, without losing the biconnectivity of the input graphs.

We claim that, in any SEFE $\langle \Gamma_1, \Gamma_2, \Gamma_3 \rangle$ of $\langle G_1, G_2, G_3 \rangle$, Property 1 and Property 2 of Lemma 8.1 hold. Observe that the claim implies the statement, since the two directions of the proof flow exactly as in Lemma 8.1. Hence, in the following we only prove that the two properties hold.

Property 1 holds for $\langle G_1, G_2, G_3 \rangle$ due to the fact that the same property holds for $\langle G_1^*, G_2^*, G_3^* \rangle$, as proved in Lemma 8.1.

As for Property 2, first note that, for each $i = 1, \ldots, m$, the subgraph of G_1 induced by vertices w_i, α_i, β_i, γ_i, and c_i is a triconnected subgraph attached to the rest of the graph through the split pair $\{w_i, \beta_i\}$. Hence, in Γ_1 the rotation scheme of c_i is either (c_i, α_i), (c_i, w_i), (c_i, γ_i), (c_i, β_i), or (c_i, α_i), (c_i, β_i), (c_i, γ_i), (c_i, w_i). This implies that in Γ_3 the rotation scheme of c_i restricted to the edges belonging to $G_\cap$ is either (c_i, α_i), (c_i, w_i), (c_i, γ_i), (c_i, β_i), or (c_i, α_i), (c_i, β_i), (c_i, γ_i), (c_i, w_i). In order to prove that x_i^α, x_i^β, and x_i^γ appear either in this or in the reverse order around u_i, note that the rotation scheme of c_i in Γ_3 is the reverse of the rotation scheme of u_i in

Γ_3 (and hence in $G_\cap$), where edges are identified based on the path between c_i and u_i in G_3 they belong to.

Finally, in order to make $G_\cap$ a spanning tree, remove edge (u_1, w_1) from $G_\cap$; add to $G_\cap$ two star graphs with 3 leaves, and add to $G_\cap$ an edge connecting u_1 to the center of the first star and an edge connecting w_1 to the center of the second star. Also, add edges to G_1, to G_2, and to G_3 among vertices of the two stars so that (i) all graphs remains biconnected, (ii) there exists an edge of G_1, an edge of G_2, and an edge of G_3 connecting a leaf of the first star to a leaf of the second star, and (iii) no edge is added to more than one graph. A suitable augmentation is shown in Fig. 8.3.

The above discussion proves the statement for $k = 3$. To extend the theorem to any value of k observe that, given an instance of SUNFLOWER SEFE with $k_0 \geq 3$ biconnected graphs whose intersection graph $G_\cap$ is a tree, an equivalent instance with $k_0 + 1$ biconnected graphs whose intersection graph is a tree can be obtained by subdividing an edge of $G_\cap$ that is not incident to a leaf with a dummy vertex and by connecting it to all the leaves of $G_\cap$ with edges only belonging to the $(k_0 + 1)$-th graph. This concludes the proof of the theorem. □

8.3 Partitioned T-Coherent k-Page Book Embedding

In this section we turn our attention to the problem of computing k-page book-embeddings in which the assignment of the k sets of edges to the k pages is given as part of the input. In the original definition of this problem [HN14], called PARTITIONED k-PAGE BOOK EMBEDDING (PBE-k), an ordering of the vertices is required such that edges belonging to the same page do not cross each other. We study a generalization of the PBE-k problem, called PARTITIONED T-COHERENT k-PAGE BOOK EMBEDDING (PTBE-k), in which the order of the vertices must satisfy an additional constraint, namely it must be represented by a tree T, also given as part of the input. Observe that, problem PTBE-k in which T is a star is exactly the same problem as PBE-k. Given that the original formulation of PBE-k is better known, we describe the results for PTBE-k when T is a star graph in terms of PBE-k.

Problem PTBE-k has been defined in [ADF+12] and proved equivalent to the case of SUNFLOWER SEFE in which the intersection graph $G_\cap$ is a spanning tree and all the edges not belonging to $G_\cap$ are incident to two leaves of such tree[2]. For this reason, in the following we will indifferently denote an instance $\langle T, E_1, \ldots, E_k \rangle$ of PTBE-k by the corresponding instance $\langle G_1, \ldots, G_k \rangle$ of SUNFLOWER SEFE, where $G_i = (V(T), E(T) \cup E_i)$, for each $i = 1, \ldots, k$, and vice versa.

[2]Although Angelini *et al.* [ADF+12] proved the equivalence only for $k = 2$, their result can be easily extended to any value of $k > 2$.

We remark that the instances of SUNFLOWER SEFE constructed in the reduction performed in Theorem 8.1 are such that the intersection graph $G_\cap$ is a spanning tree, but there exist edges not belonging to $G_\cap$ that are incident to internal vertices of such tree. In order to obtain equivalent instances of SUNFLOWER SEFE satisfying both properties, it would be possible to apply a procedure described in [ADF+12] that, for each edge $e \in \bigcup_{i=1}^{k} E_i$ incident to an internal vertex v of $G_\cap$, adds a new leaf to $G_\cap$ attached to v and replaces v with this leaf as an endvertex of e. Hence, Theorem 8.1 implies that PTBE-k is NP-complete for $k \geq 3$. However, every time a new leaf is attached to an internal vertex, such a vertex becomes a cut-vertex for $k-1$ of the input graphs; thus, none of the k graphs G_i can be assumed to be biconnected after the whole procedure has been applied.

The relevance of this latter observation is motivated by the fact that the biconnectivity of the input graphs G_i, together with the "simplicity" of T, seems to be the key factor allowing for polynomial-time algorithms for the partitioned book-embedding problems. Indeed, Hoske [Hos12] proved that PBE-k becomes solvable in linear-time if each graph G_i is *T-biconnected*, that is, E_i induces a connected graph. Notice that, T-biconnectivity is a stronger requirement than biconnectivity, since any T-biconnected graph is also biconnected, while the converse is not always true. To give an example, consider a biconnected graph composed of two copies of a complete binary tree on seven vertices whose leaves are connected by a matching. It is easy to see that such a graph cannot be spanned by a tree in such a way that the set of edges not included in the spanning tree induces a connected graph.

We observe that the algorithm by Hoske can be easily generalized from PBE-k to PTBE-k in which T is not necessarily a star; hence, the same algorithmic result can be stated also for PTBE-k. Furthermore, to support the importance of the above mentioned key factors, we recall that PTBE-k is polynomial-time solvable for $k = 2$ if either both input graphs are biconnected [BR13], or $T = G_\cap$ is a star [HN14], or $T = G_\cap$ is a binary tree [Hos12, Sch13].

In this section we provide several results that considerably narrow the gap between the instances of PTBE-k that can be solved in polynomial time and those that cannot (unless $P = NP$), by studying their complexity with respect to such factors. Namely, we prove that:

- PTBE-k remains NP-complete for $k \geq 3$ when T is a caterpillar and two of the input graphs are biconnected (Theorem 8.2);
- PTBE-k can be reduced in polynomial time to PBE-$(k+1)$ (Theorem 8.3);
- PBE-k (with no restriction on the biconnectivity of the input graphs) is NP-complete for $k \geq 3$ (Theorem 8.4), which was known only for k unbounded [Sch13];

- PTBE-k is linear-time solvable if $k-1$ of the input graphs are T-biconnected (Theorem 8.5);
- requiring one of the two graphs of an instance $\langle T, E_1, E_2\rangle$ of PTBE-2 to be a biconnected series-parallel graph does not alter the computational complexity of the problem (Theorem 8.6).

NP-completeness

Due to the equivalence between PTBE-k and SUNFLOWER SEFE in which $G_\cap$ is a spanning tree and all the edges not belonging to $G_\cap$ connect two of its leaves, in order to prove Theorem 8.2 it suffices to show that the instances produced in the reduction of Lemma 8.1 can be modified to obtain equivalent instances satisfying the above properties in which two of the input graphs are biconnected.

Theorem 8.2 PTBE-k *is NP-complete for $k \geq 3$ even if two of the input graphs are biconnected and $T = G_\cap$ is a caterpillar tree.*

Proof: Consider an instance $\langle G_1, G_2, G_3\rangle$ obtained from the reduction described in Lemma 8.1. We describe how to obtain an equivalent instance satisfying the required properties.

Refer to Fig. 8.2 and to Fig. 8.4. First, for $i = 1, \ldots, m$, replace the edges (w_i, x_i^α), (w_i, x_i^β), and (w_i, x_i^γ) of G_3 with length-2 paths composed of a black and of a green edge and such that the black edge is incident to w_i. Denote by Φ_i the star graph centered at w_i induced by the newly inserted black edges. Second, for $i = 1, \ldots, m$, subdivide edge (w_i, w_{i+1}) of $G_\cap$ (where $w_{m+1} = v_m$) with a dummy vertex t_i, and add to $G_\cap$ a star graph Ψ_i with 3 leaves centered at t_i. Observe that, at this stage of the construction, $G_\cap$ is a spanning pseudo-caterpillar.

It is now possible to obtain an equivalent instance of SUNFLOWER SEFE where G_1 and G_2 are biconnected and $G_\cap$ remains a spanning pseudo-caterpillar, by only adding edges to G_1 and to G_2 among the leaves of Φ_i and Ψ_i, for $i = 1, \ldots, m$, as in Fig. 8.4.

Further, in order to make $G_\cap$ a spanning caterpillar, remove edge (u_1, w_1) from $G_\cap$; add to $G_\cap$ two star graphs with 3 leaves, and add to $G_\cap$ an edge connecting u_1 to the center of the first star and an edge connecting w_1 to the center of the second star.

Finally, add edges to G_1, to G_2, and to G_3 among the leaves of the two stars so that (i) G_1 and G_2 are biconnected, (ii) there exists an edge of G_3 connecting a leaf of the first star to a leaf of the second star, and (iii) no edge is added to more than one graph. A suitable augmentation is shown in Fig. 8.4.

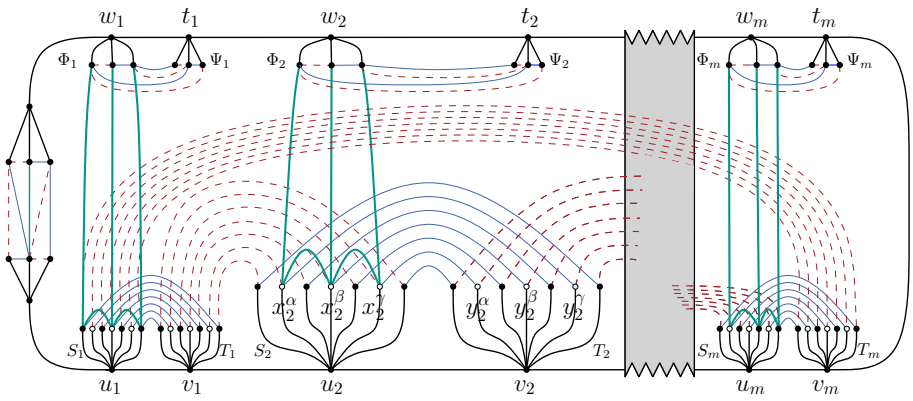


Figure 8.4: Illustration of how to modify the instance of SUNFLOWER SEFE so that: (i) the intersection graph $G_\cap$ is a spanning caterpillar and (ii) G_1 and G_2 are biconnected.

It is easy to observe that the constructed instance satisfies Properties 1 and 2 defined in Lemma 8.1. As described in the proof of the lemma, this is sufficient to prove the statement of the theorem.

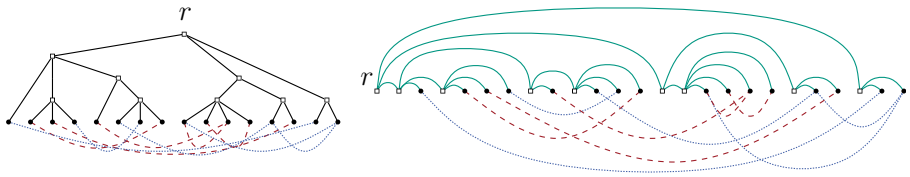
In order to extend the theorem to any value of k observe that, given an instance of SUNFLOWER SEFE with $k_0 \geq 3$ graphs satisfying the properties of the theorem, an equivalent instance with $k_0 + 1$ graphs still satisfying the properties can be obtained as follows. Subdivide an edge that is not incident to a leaf of $G_\cap$ with four dummy vertices a, b, c, d, add four new vertices a', b', c', d', and add edges (a, a'), (b, b'), (c, c'), and (d, d') to $G_\cap$. Also, add edges (a', b') and (c', d') to G_1; add edges (a', d') and (b', c') to G_2; and add edge (a', c') to the $(k_0 + 1)$-th graph. This concludes the proof of the theorem. □

In the following we prove that dropping the requirement of biconnectivity of the graphs allows us to prove NP-completeness also for PBE-k when k is bounded by a constant, thus improving on the result of Hoske [Hos12]. We first prove that the NP-completeness of PTBE-k for $k \geq 3$ proved in Theorem 8.2 implies the NP-completeness of PBE-k for $k \geq 4$. Then, in Theorem 8.4 we show that PBE-k is NP-complete even for $k = 3$. We recall that a linear-time algorithm for the problem is known when $k = 2$ [HN14].

Theorem 8.3 PTBE-k *is polynomial-time reducible to* PBE-$(k + 1)$.

Proof: Let $\langle T, E_1, \ldots, E_k \rangle$ be an instance of PTBE-k. We construct an instance $\langle V^*, E_1^*, \ldots, E_k^*, E_{k+1}^* \rangle$ of PBE-$(k+1)$ as follows.

Set $V^* = V(T)$ and $E_{k+1}^* = E(T)$. Then, for each $i = 1, \ldots, k$, set $E_i^* = E_i$. Refer to Fig. 8.5.



(a) $\langle T, E_1, E_2 \rangle$ (b) $\langle V^*, E_1^*, E_2^*, E_3^* \rangle$

Figure 8.5: Illustration of the proof of Theorem 8.3.

We prove that $\langle V^*, E_1^*, \ldots, E_k^*, E_{k+1}^* \rangle$ is a positive instance of PBE-$(k+1)$ if and only if $\langle T, E_1, \ldots, E_k \rangle$ is a positive instance of PTBE-k.

Suppose that $\langle V^*, E_1^*, \ldots, E_k^*, E_{k+1}^* \rangle$ admits a partitioned $(k+1)$-page book-embedding $\mathcal{O}^*$. Let $\mathcal{O}$ be the order obtained by restricting $\mathcal{O}^*$ to the leaves of T. We show that $\mathcal{O}$ is a partitioned T-coherent k-page book-embedding of $\langle T, E_1, \ldots, E_k \rangle$.

For each $i = 1, \ldots, k$, no two edges of E_i alternate in $\mathcal{O}$, as otherwise the corresponding two edges of E_i^* would alternate in $\mathcal{O}^*$, hence contradicting the hypothesis that $\mathcal{O}^*$ is a partitioned $(k+1)$-page book-embedding. Also, we claim that order $\mathcal{O}$ is represented by T. Namely, place the vertices of T on a horizontal line in the same order as they appear in $\mathcal{O}^*$; since $\mathcal{O}^*$ supports a crossing-free drawing of the edges of $E_{k+1}^* = E(T)$ on a single page and since $\mathcal{O}^*$ restricted to the leaves of T coincides with $\mathcal{O}$, the claim follows.

Suppose that $\langle T, E_1, \ldots, E_k \rangle$ admits a partitioned T-coherent k-page book-embedding $\mathcal{O}$. We show how to construct a partitioned $(k+1)$-page book-embedding $\mathcal{O}^*$ of $\langle V^*, E_1^*, \ldots, E_k^*, E_{k+1}^* \rangle$.

Initialize $\mathcal{O}^* = \mathcal{O}$. Root T at an arbitrary internal vertex. Then, consider each internal vertex w of T according to a bottom-up traversal. Consider the subtree $T(w)$ of T rooted at w and consider the vertex z of $T(w)$ appearing in $\mathcal{O}^*$ right before all the other vertices of $T(w)$. Place w right before z in $\mathcal{O}^*$.

We show that order $\mathcal{O}^*$ is a partitioned $(k+1)$-page book-embedding of instance $\langle V^*, E_1^*, \ldots, E_k^*, E_{k+1}^* \rangle$ of PBE-$(k+1)$.

For each $i = 1, \ldots, k$, no two edges of E_i^* alternate in $\mathcal{O}^*$, as otherwise the corresponding two edges of E_i would alternate in $\mathcal{O}$, hence contradicting the hypothesis

that $\mathcal{O}$ is a partitioned T-coherent k-page book-embedding. Also, the fact that no two edges of E^*_{k+1} alternate in $\mathcal{O}^*$ descends from the fact that, for each vertex w of T, all the vertices belonging to the subtree $T(w)$ of T rooted at w appear consecutively in $\mathcal{O}^*$. We prove this property by induction. In the base case w is the parent of a set of leaves. In this case, the statement holds since $\mathcal{O}$ is represented by T. Inductively assume that, for all children u_i of w, the vertices of $T(u_i)$ are consecutive in $\mathcal{O}^*$. By construction, w has been placed right before all vertices of $T(w)$. It follows that all vertices of $T(w)$ (including w) are consecutive in $\mathcal{O}^*$. This concludes the proof of the theorem. □

As PBE-k is a special case of PTBE-k, the problem belongs to NP. Hence, putting together the results of Theorem 8.3 and of Theorem 8.2, we obtain the following:

Corollary 8.1 PBE-k *is* NP*-complete for* $k \geq 4$.

We strengthen this result by proving that the NP-hardness of PBE-k holds even for $k = 3$. As for Theorem 8.2, we describe the proof in terms of the corresponding SUNFLOWER SEFE problem, namely in the case in which $G_\cap$ is a star graph and all the edges not belonging to $G_\cap$ connect two of its leaves.

Theorem 8.4 PBE-k *is* NP*-complete for* $k \geq 3$.

Proof: We prove the statement for $k = 3$, as for $k \geq 4$ it descends from Corollary 8.1. The NP-hardness is shown by means of a polynomial-time reduction from problem BETWEENNESS. Given an instance $\langle A, C\rangle$ of BETWEENNESS, we construct an instance $\langle G_1, G_2, G_3\rangle$ of SUNFLOWER SEFE in which $G_\cap$ is a star that admits a SEFE if and only if $\langle A, C\rangle$ is a positive instance of BETWEENNESS. Refer to Fig. 8.6.

Graph $G_\cap$ is initialized to a star graph with center ϕ, a leaf ω and, for $i = 0, \ldots, m$, leaves a_i and b_i. Also, for $i = 1, \ldots, m$, $G_\cap$ contains n leaves $x^1_i, \ldots, x^n_i$, n leaves $y^1_i, \ldots, y^n_i$, plus two additional leaves x^*_i and y^*_i. Finally, $G_\cap$ contains n leaves $y^1_0, \ldots, y^n_0$, plus an additional leaf y^*_0.

Graph G_1 contains all the edges of $G_\cap$ plus a set of edges defined as follows. For $i = 1, \ldots, m$, graph G_1 contains an edge (ω, a_i). Also, for $i = 1, \ldots, m$, graph G_1 contains edges (x^j_i, y^j_{i-1}), with $j = 1, \ldots n$, and edge (x^*_i, y^*_{i-1}).

Graph G_2 contains all the edges of $G_\cap$ plus a set of edges defined as follows. For $i = 0, \ldots, m-1$, graph G_2 contains an edge (ω, b_i). Also, for $i = 1, \ldots, m$, graph G_2 contains edges (x^j_i, y^j_i), with $j = 1, \ldots n$, and edge (x^*_i, y^*_i).

Graph G_3 contains all the edges of $G_\cap$ plus a set of edges defined as follows. Graph G_3 contains edges (ω, a_o) and (ω, b_m). Also, for each $i = 0, \ldots, m$, graph

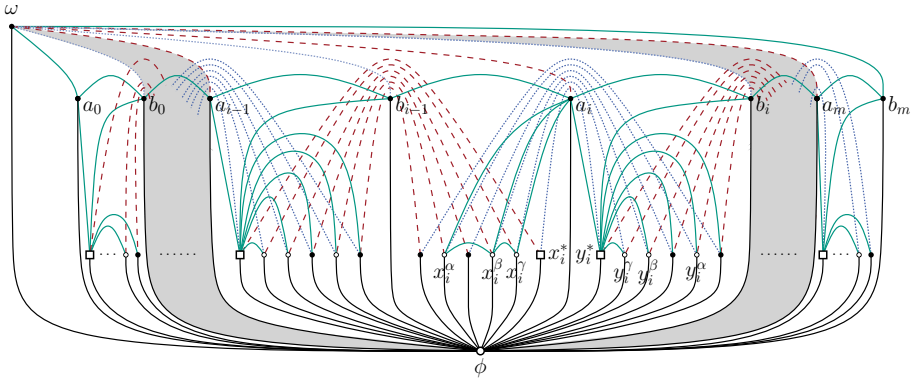


Figure 8.6: Illustration of the composition of $G_\cap$, G_1, G_2, and G_3 in Theorem 8.4, focused on the i-th triple $t_i = \langle \alpha, \beta, \gamma \rangle$ of C.

G_3 contains edges (a_i, b_i), (a_i, y_i^*), (b_i, y_i^*), and edges (y_i^*, x_i^j), with $j = 1, \ldots, n$. Finally, for $i = 1, \ldots, m$, consider the i-th triple $t_i = \langle \alpha, \beta, \gamma \rangle$ of C, and the corresponding vertices x_i^α, x_i^β, and x_i^γ; graph G_3 contains edges (a_i, x_i^α), (a_i, x_i^β), (a_i, x_i^γ), (x_i^α, x_i^β), and (x_i^β, x_i^γ).

Before proceeding with the proof that the constructed instance $\langle G_1, G_2, G_3 \rangle$ of SUNFLOWER SEFE is equivalent to instance $\langle A, C \rangle$ of BETWEENNESS, we prove some properties of $\langle G_1, G_2, G_3 \rangle$.

Namely, since vertices ϕ, ω, and vertices a_i and b_i, with $i = 1, \ldots, m$, induce a wheel with central vertex ϕ in G_3, in any planar embedding of G_3, edges (ω, ϕ), (a_o, ϕ), (b_o, ϕ),...,(a_m, ϕ), and (b_m, ϕ) appear in this order (or in the reverse order) around ϕ. Also, since y_i^* is adjacent in G_3 to both a_i and b_i, for $i = 0, \ldots, m$, edge (y_i^*, ϕ) appears between edges (a_i, ϕ) and (b_i, ϕ) around ϕ in any planar embedding of G_3. Hence, since all vertices y_i^j, with $j = 1, \ldots, n$, are adjacent in G_3 to y_i^*, also edges (y_i^j, ϕ) appear between (a_i, ϕ) and (b_i, ϕ) around ϕ in any planar embedding of G_3. Further, for $i = 1, \ldots, m$, edges (x_i^j, ϕ), with $j = 1 \ldots, n$, and edge (x_i^*, ϕ) appear between (b_{i-1}, ϕ) and (a_i, ϕ) around ϕ in $\langle \Gamma_1, \Gamma_2, \Gamma_3 \rangle$. This is due to the following two facts: (1) all vertices x_i^j and vertex x_i^* are adjacent in G_1 to a vertex y_{i-1} such that edge (y_{i-1}, ϕ) appears between edges (a_{i-1}, ϕ) and (b_{i_1}, ϕ) around ϕ, and in G_2 to a vertex y_i such that edge (y_i, ϕ) appears between edges (a_i, ϕ) and (b_i, ϕ) around ϕ; (2) there exists edges (ω, b_{i-1}) in G_2 and (ω, a_i) in G_1. Refer to Fig. 8.6 for a possible ordering of the edges around ϕ in a SEFE.

Observe that, due to the properties of the ordering of the edges of $G_\cap$ around ϕ discussed above, for $i = 1, \ldots, m$, edge (x_i^*, ϕ) and edges (x_i^j, ϕ), with $j = 1, \ldots, n$, behave similarly to the edges of the star graph S_i used in Lemma 8.1, and edge (y_i^*, ϕ) and edges (y_i^j, ϕ), with $j = 1, \ldots, n$, behave similarly to the edges of the star graph T_i used in Lemma 8.1. Namely, in any SEFE of G_1, G_2, and G_3, for each $i = 1, \ldots, m-1$, the ordering of the edges (x_i^j, ϕ), with $j = 1, \ldots, n$, and edge (x_i^*, ϕ) around ϕ is the same as the ordering of the edges (x_{i+1}^j, ϕ), with $j = 1, \ldots, n$, and edge (x_{i+1}^*, ϕ) around ϕ, where the vertices are identified based on index j. In other words, $\langle G_1, G_2, G_3 \rangle$ satisfies Property 1 of Lemma 8.1.

Further, for each $i = 1, \ldots, m$, the subgraph of G_3 induced by vertices ϕ_i, x_i^α, x_i^β, x_i^γ, and a_i is a triconnected subgraph attached to the rest of the graph through the split pair $\{\phi, a_i\}$. Hence, in any planar embedding of G_3 (and hence also in Γ_3) edges (ϕ, x_i^α), (ϕ, x_i^β), (ϕ, x_i^γ) appear either in this order or in the reverse order around ϕ. In other words, $\langle G_1, G_2, G_3 \rangle$ satisfies Property 2 of Lemma 8.1.

As described in the proof of Lemma 8.1, the fact that $\langle G_1, G_2, G_3 \rangle$ satisfies Properties 1 and 2 is sufficient to prove the statement of the theorem. □

A Polynomial-Time Algorithm

Although PTBE-k has been shown NP-complete for $k \geq 3$ even when two of the input graphs are biconnected in Theorem 8.2, we show that stronger conditions on the connectivity of the graphs allow for a polynomial-time solution of the problem. As observed before, the linear-time algorithm by Hoske [Hos12] for PBE-k with $k \geq 2$ when each graph is T-biconnected can be easily extended to solve PTBE-k under the same conditions. In the following theorem we prove that this is true even if only $k-1$ graphs are T-biconnected.

At this aim, we describe an algorithm (ALGORITHM-1) to decide whether an instance $\langle T, E_1, \ldots, E_k \rangle$ of PTBE-k is positive in the case in which $k-1$ graphs G_i are T-biconnected. In the description of the algorithm we assume, without loss of generality, that graphs $G_1, \ldots, G_{k-1}$ are T-biconnected. The algorithm consists of seven steps, described hereunder.

ALGORITHM-1:

STEP 1. For $i = 1, \ldots, k-1$, we construct an auxiliary graph H_i as follows. Initialize H_i to G_i; remove from H_i the internal vertices of T and their incident edges; and add to H_i a vertex w_i and connect it to all vertices of H_i (that is, to all leaves of T).

STEP 2. For $i = 1, \ldots, k-1$, we construct a PQ-tree $\mathcal{T}_i$ representing all possible orders of the edges around w_i in a planar embedding of H_i by applying the planarity testing algorithm of Booth and Lueker [BL76]. Since, by construction, all vertices of H_i different from w_i are adjacent to w_i, the leaves of $\mathcal{T}_i$ are in one-to-one correspondence with the leaves of T. Hence, all PQ-trees $\mathcal{T}_i$ have the same leaves.

STEP 3. We intersect all PQ-trees $\mathcal{T}_1, \ldots, \mathcal{T}_{k-1}$ to obtain a PQ-tree $\mathcal{T}^*$ representing all the possible partitioned book-embeddings of graphs $H_i \setminus w_i$, for $i = 1, \ldots, k-1$. We remark that the procedure described so far is analogous to the one described in [Hos12] to compute a PBE-k of k T-biconnected graphs.

STEP 4. We intersect[3] $\mathcal{T}^*$ with T to obtain a PQ-tree $\mathcal{T}$ representing all the possible partitioned T-coherent book-embeddings of instance $\langle T, E_1, \ldots, E_{k-1} \rangle$.

STEP 5. We construct a *representative graph* $G_{\mathcal{T}}$ from $\mathcal{T}$, as described in [FCE95b], composed of (i) *wheel* graphs (that is, graphs consisting of a cycle, called *rim*, and of a *central vertex* connected to every vertex of the rim), of (ii) edges connecting vertices of different rims not creating simple cycles containing vertices belonging to more than one wheel, and of (iii) vertices of degree 1, which are in one-to-one correspondence with the leaves of $\mathcal{T}$, each connected to a vertex of some rim.

STEP 6. We extend graph $G_{\mathcal{T}}$ by adding an edge between two degree-1 vertices if and only if the two leaves of T corresponding to such vertices are connected by an edge of E_k, hence obtaining graph H.

STEP 7. We return `YES` if H is planar, otherwise we return `NO`.

In the following theorem we prove the correctness and the time complexity of ALGORITHM-1.

Theorem 8.5 *Let $\langle T, E_1, \ldots, E_k \rangle$ be an instance of* PTBE-k *with $k \geq 2$ in which $k-1$ graphs are T-biconnected. There exists an $O(k \cdot n)$-time algorithm to decide whether $\langle T, E_1, \ldots, E_k \rangle$ admits a* PARTITIONED T-COHERENT k-PAGE BOOK EMBEDDING*, where n is the number of vertices of T.*

Proof: The algorithm that decides PTBE-k for $\langle T, E_1, \ldots, E_k \rangle$ is ALGORITHM-1.

[3] This is the extension of the algorithm by Hoske [Hos12] to instances of PTBE-k mentioned before.

We prove the correctness. First, observe that, as proved in [Hos12], the PQ-tree $\mathcal{T}^*$ constructed at STEP 3 encodes all and only the partitioned $(k-1)$-page book-embeddings of instance $\langle \mathcal{L}(T), E_1, \ldots, E_{k-1} \rangle$. Thus, intersecting $\mathcal{T}^*$ with tree T yields a PQ-tree $\mathcal{T}$ (see STEP 4) encoding all and only the partitioned T-coherent $(k-1)$-page book-embeddings of instance $\langle T, E_1, \ldots, E_{k-1} \rangle$.

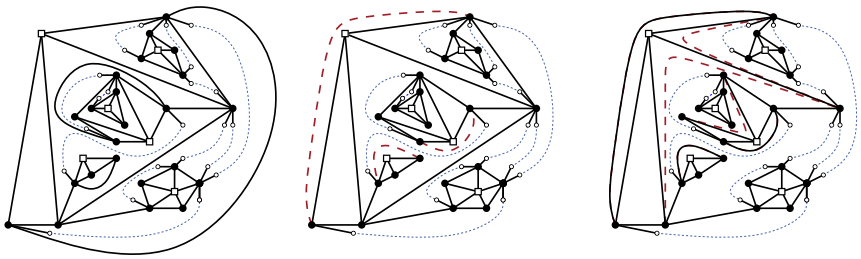
Also, as proved in [FCE95b], there exists a one-to-one correspondence between the possible orderings of the leaves of $\mathcal{T}$ and the possible orderings obtained by restricting the order of the vertices in an Eulerian tour of the outer face in a planar embedding of $G_{\mathcal{T}}$ to the degree-1 vertices.

Given a planar embedding Γ of H (see Fig. 8.7(a)), we construct a partitioned T-coherent k-page book embedding $\mathcal{O}$ of $\langle T, E_1, \ldots, E_k \rangle$. We claim that Γ can be modified in order to obtain a planar embedding Γ' of H (see Fig. 8.7(c)) such that all the degree-1 vertices of $G_{\mathcal{T}}$ lie on the outer face of the embedding $\Gamma_{\mathcal{T}}$ of $G_{\mathcal{T}}$ obtained by restricting Γ' to the vertices and edges of $G_{\mathcal{T}}$.

The claim implies that the order $\mathcal{O}$ of the degree-1 vertices in a Eulerian tour of the outer face of $\Gamma_{\mathcal{T}}$ is a partitioned T-coherent k-page book embedding of $\langle T, E_1, \ldots, E_k \rangle$ since (i) $\mathcal{O}$ is represented by $\mathcal{T}$ and (ii) no two edges of E_k alternate in $\mathcal{O}$, given that Γ' is planar.

We prove the claim. First, we show that starting from Γ we can obtain a planar drawing Γ^* of H such that every wheel of $G_{\mathcal{T}}$ is drawn *canonically* (see Fig. 8.7(b)), that is, with its central vertex lying in the interior of its rim. Consider any wheel W of $G_{\mathcal{T}}$ with central vertex ω that is not drawn canonically in Γ. This implies that there exist two vertices a and b of the rim of W such that all the vertices of W different from a, b, and ω lie in the interior of cycle (a, b, ω). Since, by construction of $G_{\mathcal{T}}$ and of H, vertex ω is not adjacent to any vertex not belonging to W, it is possible to reroute edge (a, b) as a curve arbitrarily close to path (a, ω, b) so that cycle (a, b, ω) does not enclose any vertex of H. Observe that such an operation might determine a change in the rotation scheme of a or b. Applying such a procedure to all non-canonically drawn wheels eventually results in a planar drawing Γ^* of H such that all wheels of $G_{\mathcal{T}}$ are drawn canonically. Second, we show how to obtain Γ' starting from Γ^* (see Fig. 8.7(c)). Consider any wheel W of $G_{\mathcal{T}}$, with central vertex ω. For each two adjacent vertices a and b of the rim of W, if there exist vertices of H in the interior of cycle (a, b, ω), then we reroute edge (a, b) as a curve arbitrarily close to path (a, ω, b) so that cycle (a, b, ω) does not enclose any vertex of H. Since ω is not connected to vertices of H other than those belonging to the rim of W, this operation does not introduce any crossing. After this operation has been performed for every two adjacent edges of the rim of W, there exists no vertex of H not belonging to W in the interior of the rim of W, since W is drawn canonically. This concludes the proof of the claim, since $G_{\mathcal{T}}$ does not contain any simple cycle whose vertices belong to

more than one wheel and no wheel of $G_{\mathcal{T}}$ contains in its interior vertices of H not belonging to it.



(a) (b) (c)

Figure 8.7: Illustration for the proof of Theorem 8.5. Edges of $G_{\mathcal{T}}$ are black solid curves. Edges of E_k are blue dotted curves. Edges of H which have been redrawn with respect to the previous drawing are red dashed curves. Central vertices of the wheels are white squares. Degree-1 vertices of $G_{\mathcal{T}}$ are white circles. (a) Planar drawing Γ of H. (b) Planar drawing Γ^* of H in which every wheel of $G_{\mathcal{T}}$ is drawn canonically. (c) Planar drawing Γ' of H in which all the degree-1 vertices of $G_{\mathcal{T}}$ lie in the outer face of Γ' restricted to $G_{\mathcal{T}}$.

Given a partitioned T-coherent k-page book embedding $\mathcal{O}$ of $\langle T, E_1, \ldots, E_k \rangle$, we construct a planar embedding Γ of H. To obtain Γ, we first augment $G_{\mathcal{T}}$ to an auxiliary graph U by adding a dummy edge between two degree-1 vertices of $G_{\mathcal{T}}$ if and only if the corresponding leaves of T are either adjacent in $\mathcal{O}$ or appear as the first and last element in $\mathcal{O}$. Since $\mathcal{O}$ is a partitioned T-coherent k-page book embedding $\mathcal{O}$ of $\langle T, E_1, \ldots, E_k \rangle$, it is possible to find a planar embedding of $G_{\mathcal{T}}$ in which the degree-1 vertices appear along the Eulerian tour of the outer face in the same order as $\mathcal{O}$. Hence, graph U is planar. Produce a planar drawing Γ^* of H whose outer face is the cycle composed of all the dummy edges. Since $\mathcal{O}$ is a partitioned T-coherent k-page book embedding, no two edges of E_k alternate in $\mathcal{O}$. Hence they can be drawn in the outer face of Γ^* without introducing crossings. Removing all dummy edges yields a planar embedding Γ of H.

We prove the time complexity. STEP 1 and STEP 2 take $O(k \cdot n)$ time, since the time-complexity of constructing a PQ-tree on a ground set of n elements is linear in the size of the ground set [Boo75, BL76]. STEP 3 and STEP 4 take $O((k-2) \cdot n)$ and $O(n)$ time, respectively, since the intersection of two PQ-trees can be performed in amortized linear time in their size [Boo75, BL76] and the size of the obtained PQ-tree

stays linear in the size of the ground set. STEP 5 takes linear time in the size of $\mathcal{T}$, since it corresponds to replacing each Q-node with a wheel and each P-node with a cut-vertex connecting the wheels [FCE95b]. Observe that, graph $G_{\mathcal{T}}$ has size linear in n, since each vertex of the rim of a wheel corresponds to exactly one edge of $\mathcal{T}$. STEP 6 takes $O(|E_k|) = O(n)$ time and produces a graph H with $O(n)$ vertices. Finally, testing the planarity of H takes linear time in the size of H [BL76].

This concludes the proof of the theorem. □

Partitioned T-Coherent 2-Page Book Embedding

In this subsection we restrict our attention to instances $\langle T, E_1, E_2\rangle$ of PTBE-k with $k = 2$. We remark that this problem has been proved [ADF+12] equivalent to SEFE for $k = 2$ when the intersection graph $G_\cap$ is connected. This problem was only known to be polynomial-time solvable if (i) T is a star [HN14], (ii) $G_1 = (V(T), E(T) \cup E_1)$ and $G_2 = (V(T), E(T) \cup E_2)$ are biconnected [BR13], or (iii) T is binary [Hos12, Sch13].

As a first milestone of this subsection, we observe that restricting Theorem 8.5 to the case $k = 2$ yields a result that extends the class of polynomially-solvable instances for this case.

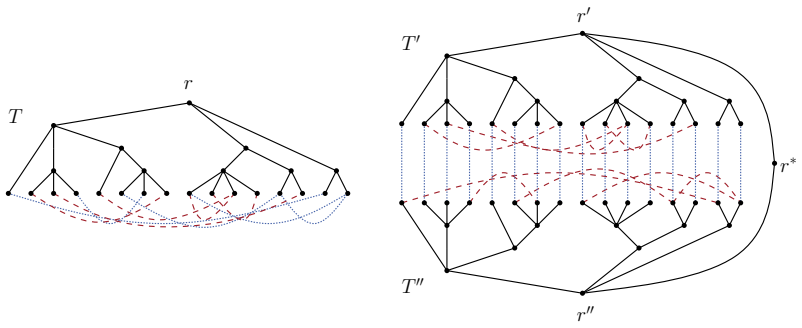
Corollary 8.2 PTBE-2 *is linear-time solvable if either G_1 or G_2 is T-biconnected.*

The second part of this subsection is instead devoted to prove that, in order to find a polynomial-time algorithm for the general setting of PTBE-2, it suffices to focus on instances of the same problem in which only one of the two graphs is biconnected (not T-biconnected) and series-parallel.

Theorem 8.6 *Let $\langle T, E_1, E_2\rangle$ be an instance of* PTBE-2. *There exists an equivalent instance $\langle T^*, E_1^*, E_2^*\rangle$ of* PTBE-2 *such that one of the two graphs is biconnected and series-parallel.*

Proof: We describe how to construct instance $\langle T^*, E_1^*, E_2^*\rangle$ starting from $\langle T, E_1, E_2\rangle$. Refer to Fig 8.8.

Let r be any internal vertex of T. Tree T^* is constructed as follows. Initialize tree T^* to the union of two copies T' and T'' of T. For each vertex $v \in T$, let v' and v'' be the two copies of v in T' and in T'', respectively. Add a vertex r^* to T^* and edges (r^*, r') and (r^*, r''). Sets E_1^* and E_2^* are defined as follows. Set $E_1^* = \{(v_i', v_j') : (v_i, v_j) \in E_1\} \cup \{(v_i'', v_j'') : (v_i, v_j) \in E_2\}$. Also, set $E_2^* = \{(v_i', v_i'') : v_i \in \mathcal{L}(T)\}$, where $\mathcal{L}(T)$ denotes the set of leaves of T.



(a) $\langle T, E_1, E_2 \rangle$ (b) $\langle T^*, E_1^*, E_2^* \rangle$

Figure 8.8: Illustration of the proof of Theorem 8.6.

It is straightforward to observe that, by construction, the graph G_2^* composed of $\mathcal{T}^*$ plus the edges in E_2^* is biconnected and series-parallel. We prove that $\langle T^*, E_1^*, E_2^* \rangle$ is equivalent to $\langle T, E_1, E_2 \rangle$.

Suppose that $\langle T, E_1, E_2 \rangle$ admits a partitioned T-coherent 2-page book embedding $\mathcal{O}$. We construct an order $\mathcal{O}^*$ for $\langle T^*, E_1^*, E_2^* \rangle$ as follows. For each $i = 1, \ldots, |\mathcal{L}(T)|$, consider the vertex v_j at position i in $\mathcal{O}$. Place vertices v'_j and v''_j at positions i and $2 \cdot |\mathcal{L}(T)| - i + 1$ in $\mathcal{O}^*$, respectively.

We prove that order $\mathcal{O}^*$ is a partitioned T-coherent 2-page book embedding of $\langle T^*, E_1^*, E_2^* \rangle$. First, we observe that $\mathcal{O}^*$ is represented by T^*, as (i) T^* is composed of two copies of T connected through r^*, (ii) $\mathcal{O}^*$ is composed of two suborders of which the first coincides with $\mathcal{O}$ and the second coincides with the reverse of $\mathcal{O}$, where each element v_j of $\mathcal{O}$ is identified with elements v'_j and v''_j of $\mathcal{O}^*$, and (iii) $\mathcal{O}$ is represented by T. Second, we prove that the endvertices of edges in E_1^* and E_2^* do not alternate in $\mathcal{O}^*$. As for the edges in E_2^*, observe that for every two edges (v'_i, v''_i) and (v'_j, v''_j) with $i < j$, both vertices v'_j and v''_j lie between v'_i and v''_i in $\mathcal{O}^*$. As for the edges in E_1^*, first observe that no alternation occurs between the endvertices of edges (v'_i, v'_j) and (v''_h, v''_k), since both v'_i and v'_j appear in $\mathcal{O}^*$ before v''_h and v''_k, by construction. Also, no two edges (v'_i, v'_j) and (v'_h, v'_k) alternate in $\mathcal{O}^*$, as otherwise edges (v_i, v_j) and (v_h, v_k) would alternate in $\mathcal{O}$. For the same reason, no two edges (v''_i, v''_j) and (v''_h, v''_k) alternate in $\mathcal{O}^*$.

Suppose that $\langle T^*, E_1^*, E_2^* \rangle$ admits a partitioned T-coherent 2-page book embedding $\mathcal{O}^*$. We first observe that in $\mathcal{O}^*$ either all vertices $v'_i \in T'$ appear consecutively or all vertices $v''_i \in T''$ do, as $\mathcal{O}^*$ is represented by T^* and T^* consists of the two

copies T' and T'' of T. Also, given a partitioned T-coherent 2-page book embedding $\mathcal{O}^1$, it is possible to obtain a new one $\mathcal{O}^2$ by performing a circular shift on the elements of $\mathcal{O}^1$, that is, by setting the first element of $\mathcal{O}^1$ as the last element of $\mathcal{O}^2$ and by setting the element at position i in $\mathcal{O}^1$ as the element at position $i-1$ in $\mathcal{O}^2$, for each $i = 2, \ldots, |\mathcal{O}^1|$. Hence, in the following, we will assume that $\mathcal{O}^*$ is such that all the vertices $v'_i \in T'$ appear before all the vertices $v''_j \in T''$.

We construct an order $\mathcal{O}$ for $\langle T, E_1, E_2 \rangle$ as follows. For each $i = 1, \ldots, |\mathcal{L}(T')|$, consider the vertex v'_j at position i in $\mathcal{O}^*$ and place vertex v_j at position i in $\mathcal{O}$.

We prove that $\mathcal{O}$ is a partitioned T-coherent 2-page book embedding of $\langle T, E_1, E_2 \rangle$. First, we observe that $\mathcal{O}$ is represented by T, as the suborder of $\mathcal{O}^*$ restricted to its first $|\mathcal{L}(T)|$ elements (that corresponds to a copy of $\mathcal{O}$) is represented by T' (that is a copy of T, where vertex $v'_i \in T'$ is identified with vertex $v_i \in T$). Second, we prove that the endvertices of edges in E_1 and E_2 do not alternate in $\mathcal{O}$. In order to prove that, first observe that the suborder $\mathcal{O}'$ of $\mathcal{O}^*$ restricted to its first $|\mathcal{L}(T)|$ elements is the reverse of the suborder $\mathcal{O}''$ of $\mathcal{O}^*$ restricted to its last $|\mathcal{L}(T)|$ elements, where vertex $v'_i \in T'$ is identified with vertex $v''_i \in T''$. This is due to the fact that (i) for every $i = 1, \ldots, |\mathcal{L}(T)|$, there exists edge (v'_i, v''_i) and (ii) all the vertices $v'_i \in T'$ appear before all the vertices $v''_j \in T''$. This implies that if the endvertices of two edges (v_i, v_j) and (v_h, v_k) belonging to E_1 (to E_2) alternate in $\mathcal{O}$, then the corresponding copies v'_i, v'_j, v'_h, and v'_k (the corresponding copies v''_i, v''_j, v''_h, and v''_k) alternate in $\mathcal{O}^*$. However, this contradicts the fact that $\mathcal{O}^*$ is a partitioned T-coherent 2-page book embedding of $\langle T, E_1, E_2 \rangle$, since edges (v'_i, v'_j) and (v'_h, v'_k) (edges (v''_i, v''_j) and (v''_h, v''_k)) exist in E_1^* by construction. This concludes the proof of the theorem. □

8.4 MAX SEFE

In this section we study the optimization version of the SEFE problem, in which two drawings of the input graphs G_1 and G_2 are searched so that as many edges of $G_\cap$ as possible are drawn the same. We study the problem in its decision version and call it MAX SEFE. Namely, given a triple $\langle G_1, G_2, k^* \rangle$ composed of two planar graphs G_1 and G_2 on the same set of vertices, and an integer $k^* > 0$, the MAX SEFE problem asks whether G_1 and G_2 admit a simultaneous embedding $\langle \Gamma_1, \Gamma_2 \rangle$ in which at most k^* edges of $G_\cap$ have a different drawing in Γ_1 and in Γ_2. First, in Lemma 8.2, we state the membership of MAX SEFE in NP, which descends from the fact that SEFE belongs to NP. Then, in Theorem 8.7 and in Theorem 8.8 we prove the NP-completeness, even under several restrictions on the input graphs G_1 and G_2, and on their intersection graph $G_\cap$.

Lemma 8.2 MAX SEFE *is in* NP.

Proof: The statement descends from the fact that the SEFE problem belongs to NP [GJP+06]. Namely, let $\langle G_1, G_2, k^* \rangle$ be an instance of MAX SEFE. Non-deterministically construct in polynomial time all the sets of at most k^* edges of $G_\cap$. Then, for each of the constructed sets, replace every edge in the set with a path of length 2 in one of the two graphs, say G_1, hence obtaining a graph G'_1, and test whether a SEFE of G'_1 and G_2 exists in polynomial time with a non-deterministic Turing machine [GJP+06]. If at least one of the performed tests succeeds, then $\langle G_1, G_2, k^* \rangle$ is a positive instance. □

In order to prove that MAX SEFE is NP-complete, we show a reduction from a variant of the NP-complete [GJ77] problem PLANAR STEINER TREE (PST), defined as follows: Given an instance $\langle G(V, E), S, k \rangle$ of PST, where $G(V, E)$ is a planar graph whose edges have weights $\omega : E \to \mathbb{N}$, $S \subset V$ is a set of *terminals*, and $k > 0$ is an integer, the PST problem asks whether there exists a tree $T^*(V^*, E^*)$ such that (1) $V^* \subseteq V$, (2) $E^* \subseteq E$, (3) $S \subseteq V^*$, and (4) $\sum_{e \in E^*} \omega(e) \leq k$. The edge weights in ω are bounded by a polynomial function $p(n)$ (see [GJ77]). In our variant, that we call UNIFORM TRIANGULATED PST (UTPST), graph G is a triangulated planar graph and all the edge weights are equal to 1.

Lemma 8.3 UNIFORM TRIANGULATED PST *is NP-complete.*

Proof: The membership in NP follows from the fact that an instance of UTPST is also an instance of PST.

The NP-hardness is proved by means of a polynomial-time reduction from the version of the PST in which all the edge weights are equal to 1, G is a subdivision of a triconnected planar graph, and no subdivision vertex is a terminal, that is known to be NP-complete [ADD+15].

Let $\langle G, S, k \rangle$ be any instance of PST with the above described properties. We construct an equivalent instance $\langle G', S', k' \rangle$ of UTPST as follows. Initialize $G' = G$. Since G' is a subdivision of a triconnected planar graph, it admits a unique planar embedding.

1. For each face f of such a planar embedding, consider the vertices $v_1, \ldots, v_h$ of f as they appear on the boundary of f. Add to G' a vertex v_f inside f and connect it to each vertex v_i, for $i = 1, \ldots, h$, with a path $v_i, u_i^1, \ldots, u_i^h, v_f$. Then, for $i = 1, \ldots, h$ and $j = 1, \ldots, h$, add to G' edge (u_i^j, u_{i+1}^j), where $h + 1 = 1$. See Fig. 8.9(a). Note that, at this stage of the construction, every face has either three or four incident vertices.

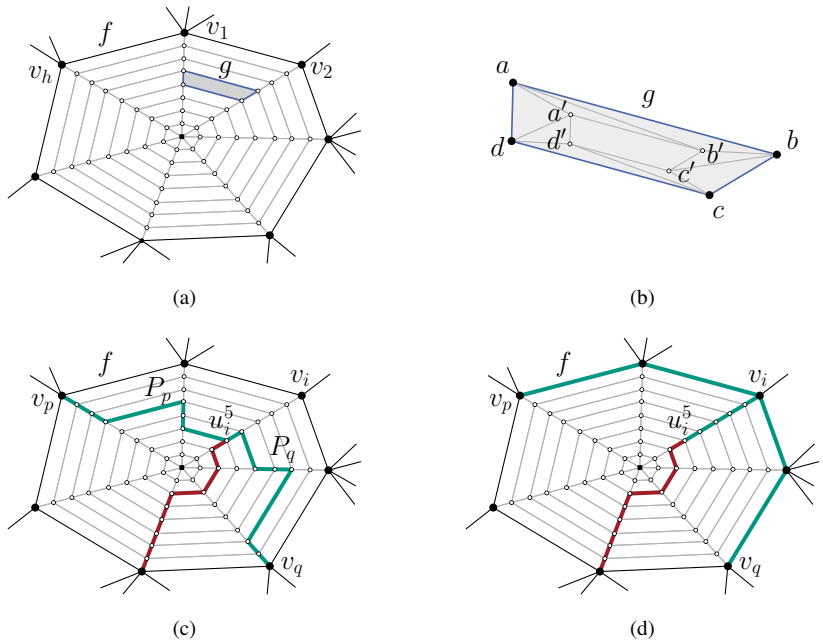


Figure 8.9: (a) Step 1 of the augmentation of a face f of graph G in the proof of Lemma 8.3. Vertex v_f is represented as a black square. (b) Step 2 of the augmentation of f, focused on a face g with 4 incident vertices. (c) The portion of T' inside f, represented by fat (red and green) edges. The green edges represent the two paths P_p and P_q connecting the degree-3 vertex u_i^j, with $j = 5$, to vertices of f. (d) The portion of T'' replacing P_p and P_q (in green).

2. For each face g with four incident vertices a, b, c, d, add inside g a 4-cycle $C_4 = (a', b', c', d')$, and add edges (a, a'), (a, b'), (b, b'), (b, c'), (c, c'), (c, d'), (d, d'), (d, a'), and (a', c'). See Fig. 8.9(b).

All the edges added in the construction have weight 1. Finally, set $S' = S$ and $k' = k$.

By construction, G' is a triangulated planar graph. Also, the reduction can be performed in polynomial time, since the number of vertices added in the construction is polynomial. In the following we prove that $\langle G, S, k\rangle$ is a positive instance of PST if and only if $\langle G', S', k'\rangle$ is a positive instance of UTPST.

Suppose that $\langle G, S, k\rangle$ is a positive instance of PST. Since $G \subseteq G'$, since none of the vertices of $G' \setminus G$ is a terminal, and since $k' = k$, the solution T of $\langle G, S, k\rangle$ is also a solution of $\langle G', S', k'\rangle$.

Suppose that $\langle G', S', k'\rangle$ is a positive instance of UTPST. Let T' be the solution of $\langle G', S', k'\rangle$. Assume that T' is an *optimal* solution of $\langle G', S', k'\rangle$, i.e., there exists no solution $T^\sharp$ of $\langle G', S', k'\rangle$ such that $\sum_{e\in T^\sharp} \omega(e) < \sum_{e\in T'} \omega(e)$.

We claim that no vertex of T' is a vertex that has been added in the interior of a face f of the unique embedding of G, that is, $T' \subseteq G$. Note that, the claim easily implies that $T = T'$ is a solution of $\langle G, S, k\rangle$.

In order to prove the claim, first observe that, for any face f of the unique embedding of G, vertex v_f does not belong to T'. This is due to the fact that no vertex in the interior of f is a terminal and, by construction, the distance between v_f and any vertex of f is $h + 1$. Hence, a tree T'' with fewer edges than T' could be obtained by replacing all the edges of T' that are in the interior of f with a set of at most $h - 1$ edges on the boundary of f.

Second, no vertex a', b', c', d' added inside a face g with four incident vertices belongs to T'. In fact, none of these vertices is a terminal, by construction; also, if T' contained one of these vertices, a tree T'' with fewer edges than T' could be obtained by replacing all the edges of T' that are in the interior of g with a set of edges on the boundary of g.

Third, no vertex of T' of degree at least 3 is a vertex that has been added in the interior of a face f of the unique embedding of G. In fact, suppose that such a vertex exists. Then, there exists a vertex u_i^j of degree at least 3, for some $1 \leq i \leq h$ and $1 \leq j \leq h$, in the interior of f with two paths P_p and P_q connecting s to two vertices v_p and v_q of f, for some $1 \leq p, q \leq h$ such that no other vertex of degree at least 3 exists along P_p and P_q or inside one of the cycles delimited by P_p, P_q, and f. See Fig. 8.9(c). In this case, a tree T'' with fewer edges than T' could be created by replacing P_p and P_q with paths $u_i^j, u_i^{j-1}, \ldots, u_i^1, v_i$ and with the shortest path along f that contains v_i, v_p, and v_q. See Fig. 8.9(d). This is due to the fact that P_p (P_q) contains at least j edges of the form (u_x^y, u_x^{y-1}) and at least $|i - p|$ edges ($|i - q|$ edges) of the form $(u_x^y, u_{x\pm1}^y)$. Hence, P_p and P_q contain at least $2j + |i-p| + |i-q|$, while the set of edges we add to T'' has size $j + m$, where $m \leq |i-p| + |i-q|$ is the length of the shortest path along f that contains v_i, v_p, and v_q.

Finally, no vertex of T' of degree 2 is a vertex that has been added in the interior of a face f of the unique embedding of G. In fact, any path connecting two vertices of f that passes through a vertex in the interior of f is longer than one of the two paths connecting such vertices along f.

This concludes the proof of the claim. As observed above, this also implies the statement of the lemma. □

Then, based on the previous lemma, we prove the first result of this section.

Theorem 8.7 MAX SEFE *is NP-complete, even if the two input graphs G_1 and G_2 are triconnected, and the intersection graph $G_\cap$ is composed of a cubic triconnected component and of a set of isolated vertices.*

Proof: The membership in NP follows from Lemma 8.2.

The NP-hardness is proved by means of a polynomial-time reduction from problem UTPST. Let $\langle G, S, k\rangle$ be an instance of UTPST. We construct an instance $\langle G_1, G_2, k^*\rangle$ of MAX SEFE as follows (refer to Fig. 8.10).

Since G is a triangulated planar graph, it admits a unique planar embedding Γ_G, up to a flip. We now construct G_1 and G_2. Initialize $G_\cap = G_1 \cap G_2$ as the dual of G with respect to Γ_G, plus a set of isolated vertices corresponding to the terminal vertices of G. Note that, the dual of G is a cubic triconnected planar graph. Consider a terminal vertex $s^* \in S$, the set $E_G(s^*)$ of the edges incident to s^* in G, and the face f_{s^*} of the dual of G composed of the edges that are dual to the edges in $E_G(s^*)$. Let v^* be any vertex incident to f_{s^*}, and let v_1^* and v_2^* be the neighbors of v^* on f_{s^*}. Subdivide edge (v^*, v_1^*) with a dummy vertex u_1^* and edge (v^*, v_2^*) with two dummy vertices u_2^* and u_3^*. Add to $G_\cap$ edges (s^*, u_1^*), (s^*, u_2^*), and (s^*, u_3^*). Since v^* has exactly one neighbor not incident to f_{s^*} in $G_\cap$, no separation pair is created. Also, vertices s^*, u_1^*, u_2^*, and u_3^* have degree 3 in $G_\cap$. Hence, $G_\cap$ remains a cubic triconnected graph plus a set of isolated vertices. See Fig. 8.10(a).

Graph G_1 contains all the vertices and edges of $G_\cap$ plus a set of edges defined as follows. For each terminal $s \in S$, consider the set $E_G(s)$ of edges incident to s in G and the face f_s of $G_\cap$ composed of the edges dual to the edges in $E_G(s)$. Add to G_1 an edge (s, v_i) for each vertex v_i incident to f_s. Note that, graph G_1 is triconnected. See Fig. 8.10(b).

Graph G_2 contains all the vertices and edges of $G_\cap$ plus a set of edges defined as follows. Rename the terminal vertices in S as $x_1, \ldots, x_{|S|}$, in such a way that $s^* = x_1$. For $i = 1, \ldots, |S| - 1$, add edge (x_i, x_{i+1}) to G_2.

In order to obtain an instance of MAX SEFE in which both graphs are triconnected, we augment G_2 to triconnected by only adding edges among vertices $\{u_1^*, u_3^*\} \cup \{x_1, \ldots, x_{|S|}\}$. See Fig. 8.10(b).

Finally, set $k^* = k$.

We show that $\langle G_1, G_2, k^*\rangle$ admits a solution if and only if $\langle G, S, k\rangle$ does.

Suppose that $\langle G, S, k\rangle$ admits a solution T. Construct a planar drawing Γ_1 of G_1. The drawing Γ_2 of G_2 is constructed as follows. The edges of $G_\cap$ that are not dual to edges of T are drawn in Γ_2 with the same curve as in Γ_1, where edges (v_1^*, u_1^*) and (v_2^*, u_3^*) play the role of (v_1^*, v^*) and (v_2^*, v^*), respectively. Observe that, in

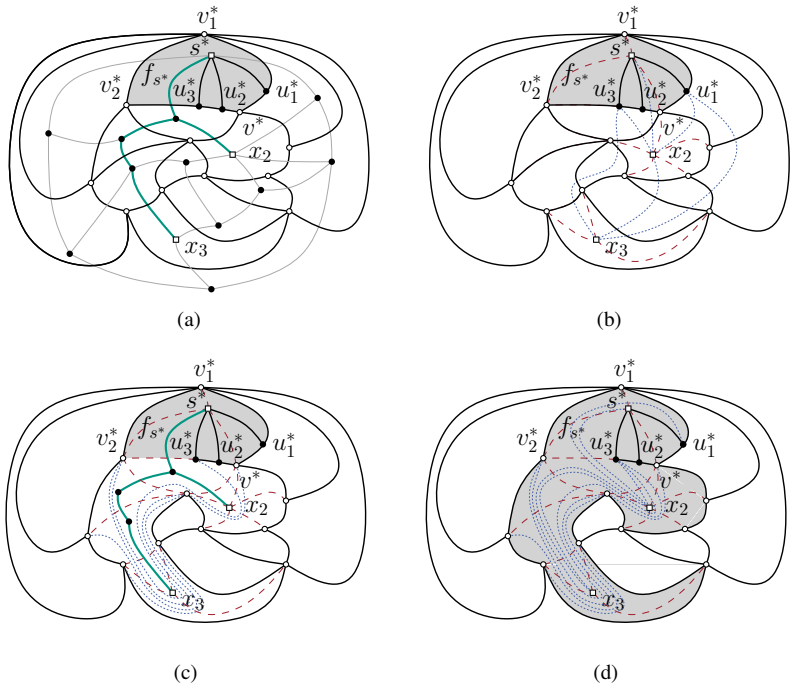


Figure 8.10: Illustration for the proof of Theorem 8.7. Black lines are edges of $G_\cap$; grey lines are edges of G; dashed red and solid blue lines are edges of G_1 and G_2, respectively; green edges compose the Steiner tree T; white squares and white circles are terminal vertices and non-terminal vertices of G, respectively. (a) $G_\cap$, G and T; (b) $G_1 \cup G_2$; (c) a drawing of $G_\cap$ where 5 edges have two different drawings; and (d) a solution $\langle \Gamma_1, \Gamma_2 \rangle$ of $\langle G_1, G_2, 5 \rangle$.

the current drawing Γ_2 all the terminal vertices in S lie inside the same face f (see Fig. 8.10(c)). Hence, all the remaining edges of G_2 can be drawn [PW01] inside f without intersections, as the subgraph of G_2 induced by the vertices incident to f and by the vertices of S is planar (see Fig. 8.10(d)). Since the only edges of $G_\cap$ that have a different drawing in Γ_1 and Γ_2 are those that are dual to edges of T, $\langle \Gamma_1, \Gamma_2 \rangle$ is a solution for $\langle G_1, G_2, k^* \rangle$.

Suppose that $\langle G_1, G_2, k^* \rangle$ admits a solution $\langle \Gamma_1, \Gamma_2 \rangle$ and assume that $\langle \Gamma_1, \Gamma_2 \rangle$ is optimal (that is, there exists no solution with fewer edges of $G_\cap$ not drawn the same). Consider the graph T composed of the dual edges of the edges of $G_\cap$ that are

not drawn the same, where edges (v_1^*, u_1^*) and (v_2^*, u_3^*) play the role of (v_1^*, v^*) and (v_2^*, v^*), respectively. We claim that T has at least one edge incident to each terminal in S and that T is connected. The claim implies that T is a solution for instance $\langle G, S, k\rangle$ of UTPST, since T has at most k edges and since $\langle \Gamma_1, \Gamma_2\rangle$ is optimal.

Suppose for a contradiction that there exist two connected components T_1 and T_2 of T (possibly composed of a single vertex). Consider the edges of G incident to vertices of T_1 and not belonging to T_1, and consider the face f_1 composed of their dual edges. Note that, f_1 is a cycle of $G_\cap$. By definition of T, all the edges incident to f_1 have the same drawing in Γ_1 and in Γ_2. Finally, there exists at least one vertex of S that lies inside f_1 and at least one that lies outside f_1. Since all the vertices in S belong to a connected subgraph of G_2 not containing any vertex incident to f_1, there exist two terminal vertices s' and s'' such that s' lies inside f_1, s'' lies outside f_1, and edge (s', s'') belongs to G_2. This implies that (s', s'') crosses an edge incident to f_1 in Γ_2, a contradiction. This concludes the proof of the theorem. □

We remark that, under the conditions on G_1, G_2, and $G_\cap$ of Theorem 8.7, the original SEFE problem is polynomial-time solvable. Actually, this is also true if each of the conditions is considered alone. Namely, it is sufficient that at least one of G_1 and G_2, say G_1, is triconnected (or has a fixed embedding) in order to have a polynomial-time algorithm [ADF+10], independently of G_2 and $G_\cap$. On the other hand, the existence of a SEFE of any two graphs G_1 and G_2 can be tested in polynomial time if $G_\cap$ is subcubic [Sch13].

In the following we go farther in the direction of investigating the complexity of MAX SEFE when the maximum degree of $G_\cap$ is bounded. In fact, we prove that MAX SEFE remains NP-complete even if the degree of the vertices in $G_\cap$ is at most 2. The proof is based on a reduction from the NP-complete problem MAX 2-XORSAT [MM11], which takes as input (i) a set of Boolean variables $B = \{x_1, ..., x_l\}$, (ii) a 2-XorSat formula $F = \bigwedge_{x_i, x_j \in B}(l_i \oplus l_j)$, where l_i is either x_i or $\overline{x_i}$ and l_j is either x_j or $\overline{x_j}$, and (iii) an integer $k > 0$, and asks whether there exists a truth assignment A for the variables in B such that at most k of the clauses in F are not satisfied by A.

Theorem 8.8 MAX SEFE *is NP-complete even if the intersection graph $G_\cap$ of the two input graphs G_1 and G_2 is composed of a set of cycles of length* 3*.*

Proof: The membership in NP follows from Lemma 8.2.

The NP-hardness is proved by means of a polynomial-time reduction from problem MAX 2-XORSAT. Let $\langle B, F, k\rangle$ be an instance of MAX 2-XORSAT. We construct an instance $\langle G_1, G_2, k^*\rangle$ of MAX SEFE as follows. Refer to Fig. 8.11(a).

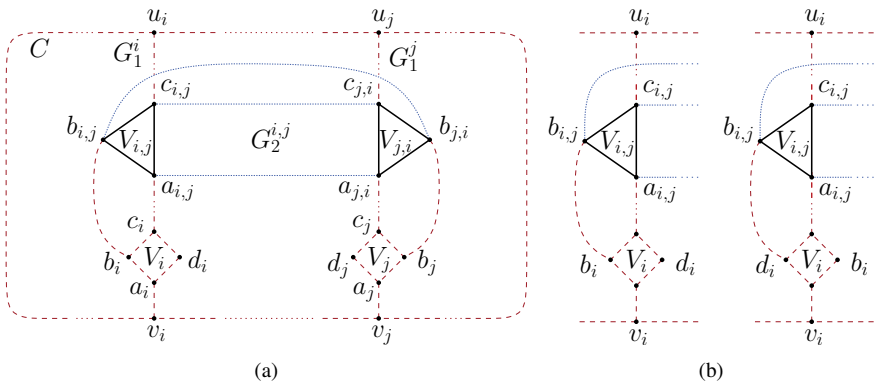


Figure 8.11: (a) Illustration of the construction of instance $\langle G_1, G_2, k^* \rangle$ of MAX SEFE. (b) Illustration of the two cases in which l_i evaluates to `true` in A.

Graph G_1 is composed of a cycle C with $2l$ vertices $v_1, v_2, \ldots, v_l, u_l, u_{l-1}, \ldots, u_1$. Also, for each variable $x_i \in B$, with $i = 1, \ldots, l$, G_1 contains a set of vertices and edges defined as follows. First, G_1 contains a 4-cycle $V_i = (a_i, b_i, c_i, d_i)$, that we call *variable gadget*, connected to C through edge (a_i, v_i). Further, for each clause $(l_i \oplus l_j) \in F$ (or $(l_j \oplus l_i) \in F$) such that $l_i \in \{x_i, \overline{x_i}\}$, G_1 contains (i) a 3-cycle $V_{i,j} = (a_{i,j}, b_{i,j}, c_{i,j})$, that we call *clause-variable gadget*, (ii) an edge $(b_{i,j}, w)$, where either $w = b_i$, if $l_i = x_i$, or $w = d_i$, if $l_i = \overline{x_i}$, and (iii) an edge $(a_{i,j}, c_{i,h})$, where $(l_i \oplus l_h)$ (or $(l_h \oplus l_i)$) is the last considered clause to which l_i participates; if $(l_i \oplus l_j)$ (or $(l_j \oplus l_i)$) is the first considered clause containing l_i, then $c_{i,h} = c_i$. When the last clause $(l_i \oplus l_q)$ (or $(l_q \oplus l_i)$) has been considered, an edge $(c_{i,q}, u_i)$ is added to G_1. Note that, the subgraph G_1^i of G_1 induced by the vertices of the variable gadget V_i and of all the clause-variable gadgets $V_{i,j}$ to which l_i participates would result in a subdivision of a triconnected planar graph by adding edge $(c_{i,q}, a_i)$, and hence it has a unique planar embedding (up to a flip). Graph G_2 is composed as follows. For each clause $(l_i \oplus l_j) \in F$, with $l_i \in \{x_i, \overline{x_i}\}$ and $l_j \in \{x_j, \overline{x_j}\}$, graph G_2 contains a triconnected graph $G_2^{i,j}$, that we call *clause gadget*, composed of all the vertices and edges of the clause-variable gadgets $V_{i,j}$ and $V_{j,i}$, plus three edges $(a_{i,j}, a_{j,i})$, $(b_{i,j}, b_{j,i})$, and $(c_{i,j}, c_{j,i})$. Finally, set $k^* = k$.

Note that, with this construction, graph $G_\cap$ is composed of a set of $2|F|$ cycles of length 3, namely the two clause-variable gadgets $V_{i,j}$ and $V_{j,i}$ for each clause $(l_i \oplus l_j)$.

We show that $\langle G_1, G_2, k^* \rangle$ admits a solution if and only if $\langle B, F, k \rangle$ does.

Suppose that $\langle B, F, k\rangle$ admits a solution, that is, an assignment A of truth values for the variables of B not satisfying at most k clauses of F. We construct a solution $\langle \Gamma_1, \Gamma_2\rangle$ of $\langle G_1, G_2, k^*\rangle$. First, we construct Γ_1. Let the face composed only of the edges of C be the outer face. For each variable x_i, with $i = 1, \ldots, l$, if x_i is `true` in A, then the rotation scheme of a_i in Γ_1 is (a_i, v_i), (a_i, b_i), (a_i, d_i) (as in Fig. 8.11(a)). Otherwise, x_i is `false` in A, and the rotation scheme of a_i is the reverse (as for a_j in Fig. 8.11(a)). Since G_1^i has a unique planar embedding, the rotation scheme of all its vertices is univocally determined. Second, we construct Γ_2. Consider each clause $(l_i \oplus l_j) \in F$, with $l_i \in \{x_i, \overline{x_i}\}$ and $l_j \in \{x_j, \overline{x_j}\}$. If l_i evaluates to `true` in A, then the embedding of $G_2^{i,j}$ is such that the rotation scheme of $a_{i,j}$ in Γ_2 is $(a_{i,j}, b_{i,j})$, $(a_{i,j}, c_{i,j})$, $(a_{i,j}, a_{j,i})$ (as in Fig. 8.11(a)). Otherwise, l_i is `false` in A and the rotation scheme of $a_{i,j}$ is the reverse (as for $a_{j,i}$ in Fig. 8.11(a)). Since $G_2^{i,j}$ is triconnected, this determines the rotation scheme of all its vertices. To obtain Γ_2, compose the embeddings of all the clause gadgets in such a way that each clause gadget lies on the outer face of all the others.

We prove that $\langle \Gamma_1, \Gamma_2\rangle$ is a solution of the MAX SEFE instance, namely that at most k^* edges of $G_\cap$ have a different drawing in Γ_1 and in Γ_2. Since $G_\cap$ is composed of 3-cycles, this corresponds to saying that at most k^* of such 3-cycles have a different embedding in Γ_1 and in Γ_2 (where the embedding of a 3-cycle is defined by the clockwise order of the vertices on its boundary). In fact, a 3-cycle with a different embedding in Γ_1 and in Γ_2 can always be realized by drawing only one of its edges with a different curve in the two drawings. By this observation and by the fact that at most $k = k^*$ clauses are not satisifed by A, the following claim is sufficient to prove the statement.

Claim 8.1 *For each clause $(l_i \oplus l_j) \in F$, if $(l_i \oplus l_j)$ is satisifed by A, then both $V_{i,j}$ and $V_{j,i}$ have the same embedding in Γ_1 and in Γ_2, while if $(l_i \oplus l_j)$ is not satisifed by A, then exactly one of them has the same embedding in Γ_1 and in Γ_2.*

Proof: Consider a clause $(l_i \oplus l_j) \in F$, where $l_i \in \{x_i, \overline{x_i}\}$ and $l_j \in \{x_j, \overline{x_j}\}$. First, we prove that $V_{i,j}$ has the same embedding in Γ_1 and in Γ_2, independently of whether $(l_i \oplus l_j)$ is satisfied or not. Namely, the flip of G_1^i selected in the construction of Γ_1 is such that the rotation scheme of $a_{i,j}$ in Γ_1 is $(a_{i,j}, b_{i,j})$, $(a_{i,j}, c_{i,j})$, $(a_{i,j}, c_x)$ if and only if l_i evaluates to `true` in A (where $c_x = c_i$ if $(l_i \oplus l_j)$ is the first considered clause involving either x_i or $\overline{x_i}$ in the construction of G_1, otherwise $c_x = c_{i,h}$ where $(l_i \oplus l_h)$ (or $(l_h \oplus l_i)$) is the clause involving either x_i or $\overline{x_i}$ considered before $(l_i \oplus l_j)$ in the construction of G_1). This can be easily verified by considering the flip of G_1^i in Γ_1 in the two cases in which l_i evaluates to `true` in A, namely when either $x_i =$ `true` and $l_i = x_i$ or when $x_i =$ `false` and $l_i = \overline{x_i}$, that are depicted in Fig. 8.11(b).

Recall that, by construction, the rotation scheme of $a_{i,j}$ in Γ_2 is $(a_{i,j}, b_{i,j})$, $(a_{i,j}, c_{i,j})$, and $(a_{i,j}, a_{j,i})$ if and only if l_i evaluates to `true` in A. Since c_x lies outside $V_{i,j}$ in Γ_1 and $a_{j,i}$ lies outside $V_{i,j}$ in Γ_2, the embedding of $V_{i,j}$ is determined by the evaluation of l_i in A in the same way in Γ_1 as in Γ_2.

Hence, it remains to prove that, if $(l_i \oplus l_j)$ is satisifed by A, then also $V_{j,i}$ has the same embedding in Γ_1 and in Γ_2. Suppose that l_j evaluates to `false` in A. By construction, the flip of G_1^j selected in the construction of Γ_1 is such that the rotation scheme of $a_{j,i}$ in Γ_1 is $(a_{j,i}, c_{j,i})$, $(a_{j,i}, b_{j,i})$, $(a_{j,i}, c_x)$ (where c_x is defined as above). This can be easily verified by considering the flip of G_1^i in Γ_1 in the two cases in which l_j evaluates to `false` in A, namely when either $x_j =$ `false` and $l_j = x_j$ or when $x_j =$ `true` and $l_j = \overline{x_j}$. Further, since $(l_i \oplus l_j)$ is satisifed by A and l_j evaluates to `false`, l_i evaluates to `true`. Hence, by construction, the rotation scheme of $a_{i,j}$ in Γ_2 is $(a_{i,j}, b_{i,j})$, $(a_{i,j}, c_{i,j})$, $(a_{i,j}, a_{j,i})$. Since $G_2^{i,j}$ is triconnected, the rotation scheme of $a_{j,i}$ in Γ_2 is $(a_{j,i}, c_{j,i})$, $(a_{j,i}, b_{j,i})$, $(a_{j,i}, a_{i,j})$. Since c_x lies outside $V_{j,i}$ in Γ_1 and $a_{i,j}$ lies outside $V_{j,i}$ in Γ_2, the embedding of $V_{j,i}$ is the same in Γ_1 and in Γ_2 when l_j evaluates to `false` in A.

The fact that the embedding of $V_{j,i}$ be the same in Γ_1 and in Γ_2 when l_j evaluates to `true` in A (and hence l_i evaluates to `false`) can be proved analogously. □

Suppose that $\langle G_1, G_2, k^* \rangle$ admits a solution $\langle \Gamma_1, \Gamma_2 \rangle$. Assume that $\langle \Gamma_1, \Gamma_2 \rangle$ is optimal, that is, there exists no solution of $\langle G_1, G_2, k^* \rangle$ with fewer edges of $G_\cap$ drawn differently. We construct a truth assignment A that is a solution of $\langle B, F, k \rangle$, as follows. For each variable x_i, with $i = 1, \ldots, l$, assign `true` to x_i if the rotation scheme of a_i in Γ_1 is (a_i, v_i), (a_i, b_i), (a_i, d_i). Otherwise, assign `false` to x_i.

We prove that A is a solution of the MAX 2-XORSAT instance, namely that at most k clauses of B are not satisfied by A. Since $\langle \Gamma_1, \Gamma_2 \rangle$ is optimal, for any 3-cycle $V_{i,j}$ of $G_\cap$, at most one edge has a different drawing in Γ_1 and in Γ_2. Also, for any clause $(l_i \oplus l_j)$, at most one of $V_{i,j}$ and $V_{j,i}$ has an edge drawn differently in Γ_1 and in Γ_2, as otherwise one could flip $G_2^{i,j}$ in Γ_2 (that is, revert the rotation scheme of all its vertices) and draw all the edges of $V_{i,j}$ and $V_{j,i}$ with the same curves as in Γ_1. Since $k = k^*$, the following claim is sufficient to prove the statement.

Claim 8.2 *For each clause gadget $G_2^{i,j}$ such that $V_{i,j}$ and $V_{j,i}$ have the same drawing in Γ_1 and in Γ_2, the corresponding clause $(l_i \oplus l_j)$ is satisfied by A.*

Proof: Consider a clause gadget $G_2^{i,j}$ and the drawing of the corresponding clause-variable gadgets $V_{i,j}$ and $V_{j,i}$ in Γ_2. Note that, since $G_2^{i,j}$ is triconnected, if the rotation scheme of $a_{i,j}$ is $(a_{i,j}, b_{i,j})$, $(a_{i,j}, c_{i,j})$, $(a_{i,j}, a_{j,i})$, then the rotation scheme of $a_{j,i}$ is $(a_{j,i}, c_{j,i})$, $(a_{j,i}, b_{j,i})$, $(a_{j,i}, a_{i,j})$. Otherwise, both the rotation schemes are reversed.

Also, consider the clause-variable gadget $V_{i,j}$ corresponding to any clause $(l_i \oplus l_j)$ or $(l_j \oplus l_i)$ involving a variable x_i. Note that, if the rotation scheme of $a_{i,j}$ in Γ_1 is $(a_{i,j}, b_{i,j})$, $(a_{i,j}, c_{i,j})$, $(a_{i,j}, c_x)$ (where c_x is defined as in the proof of Claim 8.1), then either edge $(b_{i,j}, b_i)$ exists in G_1 and the rotation scheme of a_i is (a_i, v_i), (a_i, b_i), (a_i, d_i), or edge $(b_{i,j}, d_i)$ exists in G_1 and the rotation scheme of a_i is (a_i, v_i), (a_i, d_i), (a_i, b_i). In both cases, literal l_i evaluates to `true` in A. In fact, in the former case $l_i = x_i$ and x_i is `true` in A, while in the latter case $l_i = \overline{x_i}$ and x_i is `false` in A, by the construction of $\langle G_1, G_2, k^* \rangle$ and by the assignment chosen for A. Analogously, if the rotation scheme of $a_{i,j}$ is the opposite, then l_i evaluates to `false` in A.

Consider any clause gadget $G_2^{i,j}$ such that $V_{i,j}$ and $V_{j,i}$ have the same drawing in Γ_1 and in Γ_2. By combining the observations on the relationships among the rotation schemes of the vertices belonging to the clause gadget $G_2^{i,j}$, to the clause-variable gadgets $V_{i,j}$ and $V_{j,i}$, and to the variable gadgets V_i and V_j, it is possible to conclude that l_i evaluates to `true` in A if and only if l_j evaluates to `false` in A, that is, $(l_i \oplus l_j)$ is satisfied by A. □

This concludes the proof of the theorem. □

8.5 Conclusions

In this chapter we proved several results concerning the computational complexity of some problems related to the SEFE and the PARTITIONED T-COHERENT k-PAGE BOOK EMBEDDING problems. We showed that the version of SEFE in which all graphs share the same intersection graph $G_\cap$ (SUNFLOWER SEFE) is NP-complete for $k \geq 3$ even when $G_\cap$ is a tree and all the input graphs are biconnected. This improves on the result by Schaefer [Sch13] who proved NP-completeness when $G_\cap$ is a forest of stars and two of the input graphs consist of disjoint biconnected components. Further, we prove NP-completeness of problem PTBE-k for $k \geq 3$ when T is a caterpillar and two of the input graphs are biconnected, and of problem PBE-k for $k \geq 3$. These results improve on the previously known NP-completeness for k unbounded by Hoske [Hos12]. Also, we provided a linear-time algorithm to decide PTBE-k for $k \geq 2$ when $k-1$ of the input graphs are T-biconnected. Most notably, this result enlarges the set of instances of PTBE-2 (and hence of the long-standing open problem SEFE when $G_\cap$ is connected) for which a polynomial-time algorithm is known. For this problem, we also proved that all the instances can be encoded by equivalent instances in which one of the two graphs is biconnected and series-parallel. It is also known that the biconnectivity of both the input graphs suffices to make the problem polynomial-time solvable [BKR13a]. On one hand, our results push PTBE-2 closer to the boundary of polynomiality. On the other hand, since we proved that for

$k \geq 3$ the biconnectivity of all the input graphs does not avoid NP-completeness, it is natural to wonder whether dropping the biconnectivy condition on one of the two graphs in the case $k = 2$ would make it possible to simulate the degrees of freedom that are given by the fact of having more graphs.

Moreover, we considered the optimization version MAX SEFE of SEFE with $k = 2$, in which one wants to draw as many common edges as possible with the same curve in the drawings of the two input graphs. We showed NP-completeness of this problem even under strong restrictions on the embedding of the input graphs and on the degree of the intersection graph that are sufficient to obtain polynomial-time algorithms for the original decision version of the problem.

Chapter 9

Deepening the Relationship between SEFE and C-Planarity

In this chapter[1] we deepen the understanding of the connection between the two long-standing Graph Drawing open problems that are the main subject of this thesis, that is, SEFE and C-PLANARITY.

In a recent paper at GD '12 [Sch13], Marcus Schaefer presented a reduction from C-PLANARITY to SEFE-2. We prove that a reduction exists also in the opposite direction, if we consider instances of SEFE-2 in which the graph induced by the common edges is connected (C-SEFE-2). We pose as an intriguing open question whether the two problems are polynomial-time equivalent.

9.1 Introduction

In recent years the problem of displaying together multiple relationships among the same set of entities has turned into a central subject of research in Graph Drawing and Visualization. In this context, the two major paradigms that held the stage are the *simultaneous embedding* of graphs, in which the relationships are described by means of different sets of edges among the same set of vertices, and the visualization of *clustered graphs*, in which the relationships are described by means of a single set of edges plus a cluster hierarchy grouping together vertices with semantic affinities.

We study the connection between the two main problems adhering to such paradigms, namely the SEFE and the C-PLANARITY problem.

[1] The contents of this chapter are a joint work with Patrizio Angelini, appeared in [AD14] and submitted to journal.

Due to their practical relevance and their theoretical appeal, these problems have attracted a great deal of effort in the research community. However, despite several restricted cases have been successfully settled, the question regarding the computational complexity of the original problems keeps being as elusive as it is fascinating.

In a recent work [Sch13], Marcus Schaefer leveraged the expressive power of SEFE – and in particular of SEFE-2– to generalize, in terms of polynomial-time reducibility, several graph drawing problems, including C-PLANARITY. On the other hand, also C-PLANARITY has shown a significant expressive power as it generalizes relevant problems, like STRIP PLANARITY [ADDF13a]. Most notably, two special cases of C-PLANARITY and SEFE-2 have been proved to be polynomial-time equivalent [HN14, ADF$^+$12], that is, C-PLANARITY with two clusters and SEFE-2 where the *common graph*, namely the graph composed of the edges that belong to both graphs, is a star. Motivated by such results, we pose the question whether this equivalence extends to the general case.

In this chapter we take a first step in this direction, by proving that C-SEFE-2, that is the restriction of SEFE-2 to instances in which the common graph is connected, reduces to C-PLANARITY.

Also, we give further evidence of the expressive power of C-PLANARITY by proving that this problem is able to generalize several of the graph drawing problems that can be reduced to SEFE-2. We focus in particular on the well-known problem LEVEL PLANARITY [JL02] and on two of its main variants, CLUSTERED-LEVEL PLANARITY [FB04] and T-LEVEL PLANARITY [WSP12].

The chapter is structured as follows. In Section 9.2 we give basic definitions. In Section 9.3 we show a polynomial-time reduction from C-SEFE-2 to C-PLANARITY. In Section 9.4 we give further new examples of the expressive power of C-PLANARITY. Finally, in Section 9.5 we give conclusive remarks and present some open problems.

9.2 Preliminaries

For the convenience of the reader, we recall the definition of the CONNECTED SEFE-2 (C-SEFE-2) problem and of the PARTITIONED T-COHERENT 2-PAGE BOOK EMBEDDING (PTBE-2) problem. We refer the reader to Sections 3.2 and 3.1 for further discussions and references about these problems.

Let $\langle G_1, G_2 \rangle$ be an instance of SEFE-2 such that the common graph $G_\cap = (V, E_1 \cap E_2)$ is connected. We call C-SEFE-2 problem the restriction of the SEFE-2 problem to such instances. Recall that, SEFE-2, and hence C-SEFE-2, is a special case of SUNFLOWER SEFE.

In this setting, efficient testing algorithms are known for several restricted cases. Namely, the existence of a C-SEFE-2 can be tested in polynomial-time when (i) $G_\cap$ is a star graph [HN14, ADF$^+$12, ADD12] or a subcubic graph [Sch13]; (ii) $G_\cap$ is 2-connected [HJL13, ADF$^+$12]; (iii) G_1 and G_2 are 2-connected [BR13]. However, the computational complexity of C-SEFE-2 is still open in the general case.

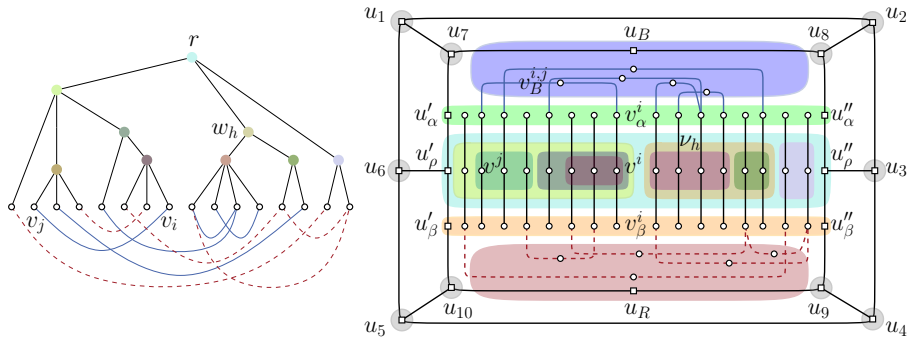
Given two planar graphs $G_1(V, E_1)$ and $G_2(V, E_2)$ such that $E_1 \cap E_2 = \emptyset$, a 2-*page book-embedding* of graphs G_1 and G_2 consists of a linear ordering $\mathcal{O}$ of the vertices of V such that for every set E_i there exist no two edges $e_1, e_2 \in E_i$ whose endvertices alternate in $\mathcal{O}$. The PTBE-2 problem takes as input a rooted tree T with leaves $\mathcal{L}(T)$ and two sets E_1, E_2 of edges among leaves such that $E_1 \cap E_2 = \emptyset$, and asks whether a 2-page book-embedding $\mathcal{O}$ of graphs $G_1 = (\mathcal{L}(T), E_1)$ and $G_2 = (\mathcal{L}(T), E_2)$ exists such that $\mathcal{O}$ is represented by T, that is, for each node v of T, the leaves of the subtree $T(v)$ of T rooted at v appear consecutively in $\mathcal{O}$. It is easy to verify that instance $\langle T, E_1, E_2\rangle$ admits a partitioned T-coherent 2-page book-embedding if and only if graphs $G_1^* = (V(T), E(T) \cup E_1)$ and $G_2^* = (V(T), E(T) \cup E_2)$ admit a SEFE. Since the reduction holds also in the opposite direction [ADF$^+$12], problems C-SEFE-2 and PTBE-2 are polynomial-time equivalent. Hence, in order to simplify the description, in the remainder of the chapter we will equivalently use the instances of the problem that better fits our scopes.

9.3 Reduction

In this section we prove the main result of the chapter, namely that C-SEFE-2 reduces to C-PLANARITY. In order to ease the description, we first present in Theorem 9.1 a reduction that produces non-flat clustered graphs; then, at the end of the section we prove in Theorem 9.2 that the reduction can be extended to obtain flat instances whose underlying graph is a set of paths.

We start by giving a high-level view of the first reduction. Due to the equivalence between C-SEFE-2 and PTBE-2 [ADF$^+$12], we can use an instance $\langle T, E_1, E_2\rangle$ of PTBE-2 to describe the reduction. Refer to Fig 9.1. In order to enforce the edges of E_1 and E_2 to lie in different pages, we subdivide each edge of these two sets and assign the subdivision vertices to two different clusters; observe that, this is analogous to the technique used in [HN14] to reduce PTBE-2 in which T is a star to C-PLANARITY. Then, we exploit a suitable cluster hierarchy to enforce the same constraints that tree T imposes on the order of the vertices along the spine of the book-embedding.

Theorem 9.1 C-SEFE-2 $\propto$ C-PLANARITY.



(a) $\langle T, E_1, E_2 \rangle$ (b) $C(G, \mathcal{T})$

Figure 9.1: Illustration of the reduction from C-SEFE-2 to C-PLANARITY. Correspondence between internal vertices of T and clusters of $\mathcal{T}$ is encoded with colors. The root cluster λ is not represented in (b).

Proof: Let $\langle T, E_1, E_2 \rangle$ be an instance of PTBE-2 (corresponding to instance $\langle G_1, G_2 \rangle$ of C-SEFE-2) and let r be the root of T. We describe how to construct an equivalent instance $C(G, \mathcal{T})$ of C-PLANARITY starting from $\langle T, E_1, E_2 \rangle$. Refer to Fig 9.1.

Initialize the underlying graph G to a graph H composed of two cycles $C_1 = \langle u_1, u_2, u_3, u_4, u_5, u_6 \rangle$ and $C_2 = \langle u_7, u_B, u_8, u''_\alpha, u''_\rho, u''_\beta, u_9, u_R, u_{10}, u'_\beta, u'_\rho, u'_\alpha \rangle$, and of edges (u_1, u_7), (u_2, u_8), (u_3, u''_ρ), (u_4, u_9), (u_5, u_{10}), and (u_6, u'_ρ). Observe that H is a subdivision of a 3-connected planar graph.

Initialize $\mathcal{T}$ to a tree only composed of a root λ. For $m = 1, \ldots, 10$, add a cluster μ_m to $\mathcal{T}$ as a child of λ, containing only vertex u_m. Also, add clusters μ_B and μ_R to $\mathcal{T}$ as children of λ, containing vertices u_B and u_R, respectively. Finally, for $\sigma \in \{\alpha, \rho, \beta\}$, add a cluster μ_σ to $\mathcal{T}$ as a child of λ, containing vertices u'_σ and u''_σ.

Then, consider each internal vertex w_h of T according to a top-down traversal of T and add to $\mathcal{T}$ a cluster ν_h either as a child of cluster ν_k, if $w_k \neq r$ is the parent of w_h in T, or as a child of cluster μ_ρ, if r is the parent of w_h in T. Also, for each leaf vertex v_i of T, add to G a path $(v^i_\alpha, v^i, v^i_\beta)$, that we call *leaf-path*. Add vertices v^i_α and v^i_β to clusters μ_α and μ_β, respectively; add v^i to cluster ν_h, if w_h is the parent of v_i in T, or to cluster μ_ρ, if r is the parent of v_i in T.

Finally, for each edge (v_i, v_j) in E_1 or in E_2, add to G path $(v^i_\beta, v^{i,j}_R, v^j_\beta)$ or path $(v^i_\alpha, v^{i,j}_B, v^j_\alpha)$, respectively, that we call *edge-paths*. Add each vertex $v^{i,j}_R$ to μ_R and

each vertex $v_B^{i,j}$ to μ_B.

Suppose that $\langle T, E_1, E_2\rangle$ admits a SEFE $\langle \Gamma_1, \Gamma_2\rangle$. We show how to construct a c-planar drawing Γ of $C(G, \mathcal{T})$. We will construct the drawing of G contained in Γ as a straight-line drawing; hence, we only describe how to place the vertices of G. Refer to Fig. 9.2.

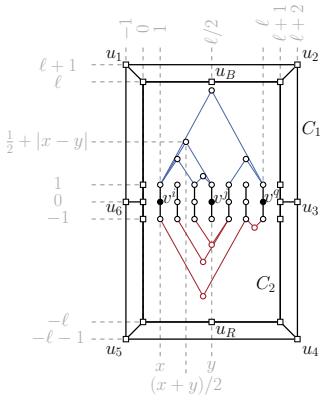


Figure 9.2: Construction of a c-planar drawing of $C(G, \mathcal{T})$ starting from $\langle \Gamma_1, \Gamma_2\rangle$, where $x = \phi(v_i)$ and $y = \phi(v_j)$.

Let $\ell = |\mathcal{L}(T)|$. We first consider cycle C_1. Place vertex u_1 at point $(-1, \ell+1)$, u_2 at $(\ell+2, \ell+1)$, u_3 at $(\ell+2, 0)$, u_4 at $(\ell+2, -\ell-1)$, u_5 at $(-1, -\ell-1)$, and u_6 at $(-1, 0)$. Then, we consider cycle C_2. Place vertex u_7 at point $(0, \ell)$, u_B at $(\frac{\ell}{2}, \ell)$, u_8 at $(\ell+1, \ell)$, u''_α at $(\ell+1, 1)$, u''_ρ at $(\ell+1, 0)$, u''_β at $(\ell+1, -1)$, u_9 at $(\ell+1, -\ell)$, u_R at $(\frac{\ell}{2}, -\ell)$, u_{10} at $(0, -\ell)$, u'_β at $(0, -1)$, u'_ρ at $(0, 0)$, and u'_α at $(0, 1)$.

Consider the circular order of the leaves of T determined by $\langle \Gamma_1, \Gamma_2\rangle$ and consider two consecutive leaves v' and v'' such that the lowest common ancestor of v' and v'' in T is the root r (note that, if r has degree greater than 1, there always exist two such vertices; otherwise, we can obtain an equivalent instance of SEFE by removing r from T). Consider the linear order $\mathcal{O}$ of the leaves of T obtained from this circular order by selecting v' and v'' as the first and the last element of $\mathcal{O}$, respectively. Let $\phi : \mathcal{L}(T) \to 1, \ldots, \ell$ be a function such that $\phi(v_i) = k$ if v_i is the k-th element in $\mathcal{O}$. For each leaf vertex v_i, we draw leaf-path $(v^i_\alpha, v^i, v^i_\beta)$ by placing vertex v^i at point $(x, 0)$, v^i_α at $(x, 1)$, and v^i_β at $(x, -1)$, where $x = \phi(v_i)$. Then, for each edge $(v_i, v_j) \in E_2$, we draw edge-path $(v^i, v_B^{i,j}, v^j)$ by placing vertex $v_B^{i,j}$ at point $(\frac{x+y}{2}, |x-y| + \frac{1}{2})$, where $x = \phi(v_i)$ and $y = \phi(v_j)$. Symmetrically, for each

edge $(v_i, v_j) \in E_1$, we draw edge-path $(v^i, v_R^{i,j}, v^j)$ by placing vertex $v_R^{i,j}$ at point $(\frac{x+y}{2}, -|x-y| - \frac{1}{2})$, where $x = \phi(v_i)$ and $y = \phi(v_j)$.

Finally, we draw the region representing each cluster. Consider each cluster $\mu \in \mathcal{T}$ according to a bottom-up traversal and draw μ as an axis-parallel rectangular region enclosing all and only the vertices and clusters in the subtree of $\mathcal{T}$ rooted at μ. Observe that, this is always possible. Namely, for clusters μ_m, with $m = 1, \ldots, 10$, and clusters μ_B, μ_R, μ_α, and μ_β this directly follows from the construction. Also, for each cluster ν_h corresponding to an internal vertex w_h of T, this descends from the fact that the ordering of the leaves of T is determined by a SEFE $\langle \Gamma_1, \Gamma_2 \rangle$. Indeed, since the drawing of T is planar in $\langle \Gamma_1, \Gamma_2 \rangle$, for any two vertices v^i and v^j of G belonging to the same cluster, there exists no vertex v^k with $\phi(v_i) < \phi(v_k) < \phi(v_j)$ belonging to a different cluster. Since all leaf-paths are drawn as vertical segments, this implies that no edge-region crossing occurs between a cluster ν_h and an edge of a leaf-path. Once all clusters ν_h have been drawn, cluster μ_ρ can be drawn to enclose all and only such clusters.

Further, observe that there exist no two edge-paths $(v_\alpha^i, v_B^{i,j}, v_\alpha^j)$ and $(v_\alpha^p, v_B^{p,q}, v_\alpha^q)$, corresponding to edges (v_i, v_j) and (v_p, v_q) of E_2, such that pairs $\langle v_i, v_j \rangle$ and $\langle v_p, v_q \rangle$ alternate in $\mathcal{O}$. Hence, any two edge-paths are either disjoint or nested. In both cases, by construction, they do not cross (see Fig. 9.2 for an illustration of the two cases). In particular, if they are disjoint the statement is trivially true; if they are nested, we only describe the limit case in which one of the endvertices of the two nesting edge-paths $(v_\alpha^i, v_B^{i,j}, v_\alpha^j)$ and $(v_\alpha^p, v_B^{p,q}, v_\alpha^q)$ coincide, the other case being analogous. Assume without loss of generality that $i = p$ and $\phi(v_i) < \phi(v_j) < \phi(v_q)$. We compute the slope coefficient of edge $(v_\alpha^i, v_B^{i,j})$.

$$\frac{y(v_B^{i,j}) - y(v_\alpha^i)}{x(v_B^{i,j}) - x(v_\alpha^i)} = \frac{(|\phi(j) - \phi(i)| + \frac{1}{2}) - 1}{|\phi(j) - \phi(i)|/2} = \frac{2|\phi(j) - \phi(i)| - 1}{|\phi(j) - \phi(i)|} = 2 - \frac{1}{|\phi(j) - \phi(i)|}$$

Analogously, the slope coefficient of edge $(v_\alpha^i, v_B^{i,q})$ is $2 - \frac{1}{|\phi(q) - \phi(i)|}$. Since $|\phi(j) - \phi(i)| < |\phi(q) - \phi(i)|$ and since $x(v_B^{i,j}) < x(v_B^{i,q})$, the statement follows. The fact that no two edge-paths corresponding to edges of E_1 cross can be proved in the same way. This concludes the proof that Γ is a c-planar drawing of $C(G, \mathcal{T})$.

Suppose that $C(G, \mathcal{T})$ admits a c-planar drawing Γ. We show how to construct a SEFE $\langle \Gamma_1, \Gamma_2 \rangle$ of $\langle T, E_1, E_2 \rangle$. First, observe that all leaf-paths entirely lie inside the face f of H delimited by cycle C_2, as f is the only face of H shared by u'_α, u'_ρ, u'_β, u''_α, u''_ρ, and u''_β. Since all vertices $v_B^{i,j}$ and $v_R^{i,j}$ are adjacent to vertices of leaf-paths, they also lie inside f. Further, since for $\sigma \in \{\alpha, \rho, \beta\}$ cluster μ_σ is represented by

a connected region enclosing vertices u'_σ and u''_σ and not involved in any edge-region and region-region crossing, all the edges connecting vertices of μ_σ to vertices of the same cluster are consecutive in the order of the edges crossing the boundary of μ_σ. This implies that the order in which leaf-paths cross the boundary of μ_α is the reverse of the order in which they cross the boundary of μ_β, since no two leaf-paths cross each other in Γ. To obtain $\langle \Gamma_1, \Gamma_2 \rangle$, we order the leaves v_i of T according to the order in which leaf-paths cross the boundary of μ_α. Let $\mathcal{O}$ be such an order.

First, we show that $\mathcal{O}$ is represented by T. In fact, by construction, for each internal vertex w_h of T, the leaves of the subtree $T(w_h)$ of T rooted at w_h belong to the same cluster ν_h. Also, since Γ is c-planar, all the leaf-paths $(v^i_\alpha, v^i, v^i_\beta)$ such that v_i is a leaf of $T(w_h)$ are consecutive in the order in which leaf-paths cross the boundary of μ_α and hence the corresponding leaves v_i are consecutive in $\mathcal{O}$. Second, we show how to construct two planar drawings Γ_1 and Γ_2 of G_1 and G_2, respectively, such that the drawing of T contained in Γ_1 and in Γ_2 coincides with Γ_T. We describe the algorithm to construct Γ_2, the algorithm for Γ_1 being analogous. Consider two edges (v_i, v_j) and (v_p, v_q) of E_2. Since the drawing of G in Γ is planar, the corresponding edge-paths $(v^i_\alpha, v^{i,j}_B, v^j_\alpha)$ and $(v^p_\alpha, v^{p,q}_B, v^q_\alpha)$ do not intersect in Γ. Also, since the edges belonging to edge-paths are consecutive in the order in which edges incident to vertices of μ_α cross the boundary of μ_α, the pair of leaves $\langle v_i, v_j \rangle$ and $\langle v_p, v_q \rangle$ of T corresponding to vertices v^i_α, v^j_α, v^p_α, and v^q_α do not alternate in $\mathcal{O}$. Hence, Γ_2 can be obtained from Γ_T by drawing the edges of E_2 as curves intersecting neither edges of T nor other edges in E_2. Since the drawing of $G_\cap = T$ is the same in Γ_1 and in Γ_2, $\langle \Gamma_1, \Gamma_2 \rangle$ is a SEFE of $\langle T, E_1, E_2 \rangle$. This concludes the proof of the theorem. □

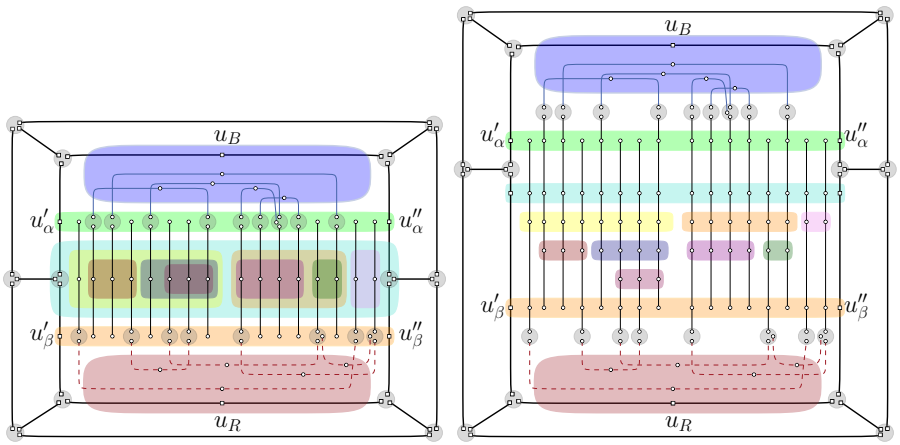
In the following we prove that the reduction of Theorem 9.1 can be modified in such a way that the resulting instance of C-PLANARITY is flat and the underlying graph consists of a set of paths.

Theorem 9.2 C-SEFE-2 $\propto$ C-PLANARITY *with flat cluster hierarchy and underlying graph that is a set of paths.*

Proof: Let $\langle T, E_1, E_2 \rangle$ be an instance of C-SEFE-2. We describe how to construct an equivalent instance $C(G, \mathcal{T})$ of C-PLANARITY with flat cluster hierarchy and underlying graph that is a set of paths starting from $\langle T, E_1, E_2 \rangle$.

First, we construct an instance $C^*(G^*, \mathcal{T}^*)$ of C-PLANARITY with non-flat cluster hierarchy by applying the reduction shown in Theorem 9.1. We describe how to transform C^* into an equivalent instance $C(G, \mathcal{T})$ of C-PLANARITY with the required properties.

For vertices u'_ρ, u''_ρ, and for all vertices v^i_α and v^i_β having degree at least 2, consider the parent cluster ν of any such vertex v in $\mathcal{T}$. Add a cluster μ_v to $\mathcal{T}$ as a child of



(a) $C''(G'', \mathcal{T}'')$ (b) $C(G, \mathcal{T})$

Figure 9.3: Construction of an equivalent instance $C(G, \mathcal{T})$ with the desired properties. (a) Obtaining an instance whose underlying graph is a set of paths. (b) Obtaining a flat instance.

ν and containing only vertex v. The obtained instance $C'(G' = G, \mathcal{T}')$ is obviously equivalent to C^*.

Let Δ be the set of all clusters $\tau \in \mathcal{T}'$ such that $\mathcal{T}'(\tau)$ has only one leaf t. Note that, Δ consists of all clusters μ_m, with $m = 1, \ldots, 10$, and all clusters added at the previous step to obtain C'. For each cluster $\tau \in \Delta$, we perform the following procedure. For each edge (t, z) of G' such that $t \in \tau$, add a vertex t_z to τ and add edge (t_z, z) to G'. Finally, remove vertex t and its incident edges from C'. This can be seen as replacing t with $\deg(t)$ copies of it. For simplicity, in the following we keep the same notation $(v^i_\alpha, v^i, v^i_\beta)$ for leaf-paths, and $(v^i_\alpha, v^{i,j}_B, v^j_\alpha)$ and $(v^i_\beta, v^{i,j}_R, v^j_\beta)$ for edge-paths, where their endvertices have been naturally replaced by the appropriate copy. See Fig. 9.3(a) for an illustration of this step.

Observe that the constructed instance $C''(G'', \mathcal{T}'')$ is such that G'' consists of a set of paths. In fact, after performing the two steps described above, each vertex of G'' has either degree 1 or degree 2. Also, every vertex of degree 2 is the middle vertex of either a leaf-path or an edge-path. Hence, no cycle is created. Further, $C''(G'', \mathcal{T}'')$ is equivalent to C', as in any c-planar drawing of C'' a vertex t that has been removed from a cluster τ can be reinserted inside $R(\tau)$ and connected to all the vertices of τ

while maintaining c-planarity (the other direction being trivial).

We now show how to construct instance C starting from C''. For each vertex v^i_α whose parent τ^i_α in $\mathcal{T}''$ is different from μ_α, we subdivide edge (v^i_α, v^i) with a vertex z^i_α; we add z^i_α to cluster μ_α; and we remove τ^i_α from the children of μ_α and add τ^i_α as a child of the root λ. For each vertex v^i_β whose parent τ^i_β in $\mathcal{T}''$ is different from μ_β, we subdivide edge (v^i_β, v^i) with a vertex z^i_β; we add z^i_β to cluster μ_β; and we remove τ^i_β from the children of μ_β and add τ^i_β as a child of the root λ.

Let μ' and μ'' be the parent clusters of the 3 copies of u'_ρ and u''_ρ, respectively, in $\mathcal{T}''$. Subdivide the edge connecting a vertex in μ' to u'_β with a new vertex and the edge connecting a vertex in μ'' to u''_β with a new vertex, and add both such vertices to μ_ρ. Also, remove μ' and μ'' from the children of μ_ρ and add them as children of the root λ.

Further, as long as there exists a cluster $\mu \neq \mu_\rho \in \mathcal{T}''(\mu_\rho)$ such that all the children of μ are leaves, we perform the following procedure. We add a new cluster μ' to $\mathcal{T}''$ as a child of the root λ. Consider the parent ν of μ in $\mathcal{T}''$. For each vertex $v^i \in \mu$, we remove v^i from the children of μ and add it as a child of ν; also, we subdivide the unique edge (v^i_β, x) incident to v^i_β with a new vertex that we add to cluster μ'. Finally, we remove μ from $\mathcal{T}''$. The instance $C(G,T)$ obtained by applying the reduction to the C-SEFE-2 instance of Fig. 9.1(a) can be seen in Fig. 9.3(b). In order to prove that C is equivalent to C'', observe that paths $(v^i_\alpha, z^i_\alpha, v^i, z^i_\beta, \ldots, v^i_\beta)$ obtained from leaf-paths $(v^i_\alpha, v^i, v^i_\beta)$ cross the boundary of $R(\mu_\alpha)$ in C in the same order in which the corresponding leaf-paths cross the boundary of $R(\mu_\alpha)$ in C''. Namely, for each cluster $\mu \in \mathcal{T}''$ there exists a cluster $\mu' \in \mathcal{T}$ imposing the same consecutivity constraint on the ordering in which paths cross the boundary of $R(\mu_\alpha)$. This concludes the proof of the theorem. □

We remark that the same technique used in Theorem 9.1 can be used to transform any instance of C-PLANARITY into an equivalent instance in which the underlying graph is a set of paths (actually, a matching). However, in general, this results in an instance whose cluster hierarchy is non-flat, even if the original instance had a flat cluster hierarchy.

Given the equivalence between C-SEFE-2 and PTBE-2, we can extend the result of Theorem 9.1 as follows.

Corollary 9.1 PTBE-2 $\propto$ C-PLANARITY *with flat cluster hierarchy and underlying graph that is a set of paths.*

9.4 The Expressive Power of C-Planarity

In this section we further motivate the question we pose in this chapter about the possible equivalence between C-PLANARITY and SEFE-2, by providing other examples of the expressive power of C-PLANARITY. Namely, we study the reducibility to C-PLANARITY of some well-known constrained-planarity problems that have been shown to be reducible to SEFE-2. Figure 0.1 in the Introduction provides an overview of the known relationships among several constrained-planarity problems, as first depicted by Schaefer in [Sch13] and updated according to the results of this thesis. See Fig. 9.4 in which the contributions of the chapters of Parts IV are highlighted.

In the following we recall the definition of problems LEVEL PLANARITY, CL-PLANARITY, and T-LEVEL PLANARITY (see Chapter 7), and problem STRIP PLANARITY (see Chapter 6), whose reducibility to C-PLANARITY will be discussed in the following.

A *level graph* $G = (V, E, \gamma)$ is a graph with an assignment γ of the vertices in V to k horizontal lines $L_1, \ldots, L_k$ with $1 \leq k \leq |V|$, called *levels*, such that no two vertices assigned to the same level are connected by an edge. A level graph $G = (V, E, \gamma)$ is *proper* if every edge spans two consecutive levels. A *level planar drawing* of (V, E, γ) maps each vertex v to a point on the line $L_i : y = i$, and each edge to a y-monotone curve between its endpoints so that no two edges intersect. The problem of testing the existence of a level planar drawing of a level graph is the LEVEL PLANARITY problem.

A *clustered-level graph* (V, E, γ, T) is a level graph (V, E, γ) equipped with a cluster hierarchy T over its vertices. A *clustered-level planar drawing* of (V, E, γ, T) is a level planar drawing of (V, E, γ) together with a representation of each cluster μ as a simple region enclosing all and only the vertices in V_μ such that: (i) no edge intersects the boundary of a cluster more than once; (ii) no two cluster boundaries intersect; and (iii) the intersection of L_i with any cluster μ is a straight-line segment, that is, the vertices of level L_i that belong to μ are consecutive along L_i. The problem of testing the existence of a clustered-level planar drawing of a clustered-level graph is the CL-PLANARITY problem.

A *$\mathcal{T}$-level graph* $(V, E, \gamma, \mathcal{T})$ is a level graph (V, E, γ) equipped with a set $\mathcal{T} = T_1, \ldots, T_k$ of trees such that the leaves of T_i are the vertices of level L_i, for $1 \leq i \leq k$. A *$\mathcal{T}$-level planar drawing* of $(V, E, \gamma, \mathcal{T})$ is a level planar drawing of (V, E, γ) such that, for $i = 1, \ldots, k$, the order in which the vertices of L_i appear along L_i is represented by T_i. The problem of testing the existence of a $\mathcal{T}$-level planar drawing of a $\mathcal{T}$-level graph is the T-LEVEL PLANARITY problem.

Hereafter, we refer to the variants of the CL-PLANARITY problem and of the T-LEVEL PLANARITY problem in which the underlying level graph is proper as the

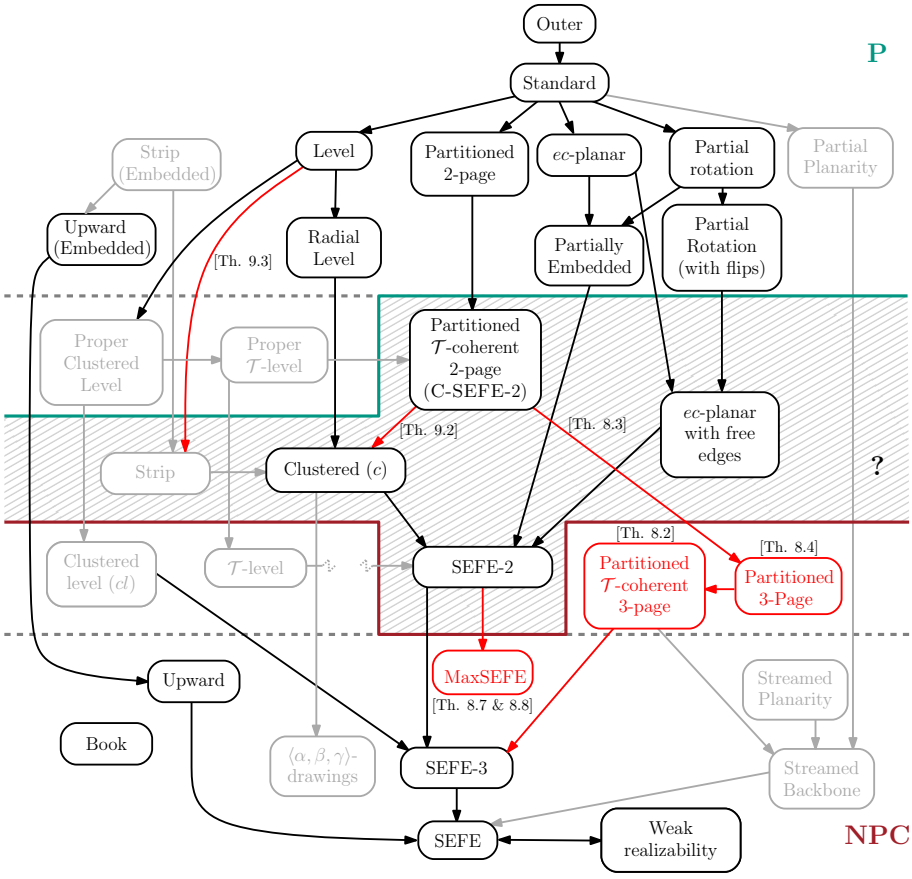


Figure 9.4: View of the schema proposed in the Introduction in which the contributions of Chapters 8 and 9 of Part IV are highlighted by using the red color.

PROPER CL-PLANARITY problem and as the PROPER T-LEVEL PLANARITY problem, respectively.

A *strip graph* $G = (V, E, \gamma)$ is a graph with an assignment γ of the vertices in V to k axis-aligned rectangular regions $R_1, \ldots, R_k$ with $1 \leq k \leq |V|$, called *strips*, such that strip R_i lies entirely above region R_{i+1}, for each $i = 1, \ldots, k-1$. A *strip planar drawing* of (V, E, γ) maps each vertex v to a point inside the strip R_i it is assigned to, and each edge to a y-monotone curve between its endpoints so that no

two edges intersect. The problem of testing the existence of a strip planar drawing of a strip graph is the STRIP PLANARITY problem.

Since LEVEL PLANARITY, PROPER T-LEVEL PLANARITY, and PROPER CL-PLANARITY can be reduced in polynomial time to C-SEFE-2, see [Sch13, ALD+14b] and Chapter 7, the result we proved in Theorem 9.2 implies the following.

Corollary 9.2 LEVEL PLANARITY, PROPER T-LEVEL PLANARITY, *and* PROPER CL-PLANARITY $\propto$ C-PLANARITY *with flat cluster hierarchy and underlying graph that is a set of paths.*

For STRIP PLANARITY, instead, no reduction to C-SEFE-2 is known, to the best of our knowledge. However, it is interesting to note that a reduction from STRIP PLANARITY to C-PLANARITY (and hence to SEFE-2) exists, as proved in Chapter 6 (refer also to [ADDF13b]). All the more so, in a recent paper by Fulek [Ful14], this result has been strengthened by showing a reduction yielding instances with flat cluster hierarchy consisting of only three clusters.

We observe that a direct reduction from PROPER T-LEVEL PLANARITY (and hence from LEVEL PLANARITY) to C-PLANARITY (with flat cluster hierarchy and underlying graph that is a set of paths), not exploiting the transformation to C-SEFE-2, can be obtained with similar techniques as those we used in the proofs of Theorems 9.1 and 9.2. In fact, for each level L_i of the level-graph, we introduce a leaf-path for each of the vertices of level L_i. The three vertices of the leaf-path are then assigned to three clusters that correspond to clusters μ_α, μ_β, and μ_ρ of Theorem 9.1. Moreover, a cluster hierarchy representing the constraints imposed by the tree T_i of L_i is added to the middle vertices of the leaf-paths as a child of μ_ρ. Finally, for each edge that connects two vertices v_j and v_h at level L_i and L_{i+1}, respectively, add an edge between the extremal vertex of the leaf-path corresponding to v_j in μ_β and the extremal vertex of the leaf-path corresponding to v_h in μ_α. See Fig. 9.5.

Note that both the direct reduction and the one via C-SEFE-2 yield instances of C-PLANARITY in which the number of clusters is linear in the size of the original instance of PROPER T-LEVEL PLANARITY. For LEVEL PLANARITY, instead, a reduction to C-PLANARITY yielding instances with flat cluster hierarchy consisting of only three clusters can be obtained by reducing it to STRIP PLANARITY, which in turn can be reduced to C-PLANARITY where the produced instances have this property [Ful14].

A reduction from LEVEL PLANARITY to STRIP PLANARITY can be easily obtained as follows. Assume that the instance of LEVEL PLANARITY is proper, as otherwise it can be made proper by adding dummy vertices along the edges spanning more than two levels. First, for each level L_i, assign all the vertices of L_i to strip

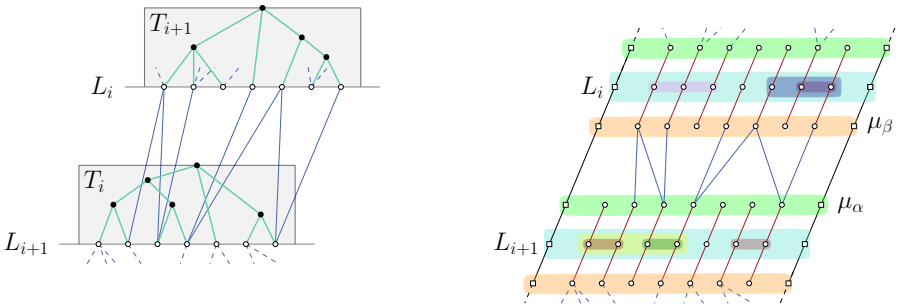


Figure 9.5: Illustration of the reduction from PROPER T-LEVEL PLANARITY to C-PLANARITY, focused on two consecutive levels L_i and L_{i+1}.

R_{3i-2}. Then, subdivide each edge connecting a vertex u in L_i to a vertex v in L_{i+1} with two dummy vertices u' and v', and assign u' to strip R_{3i-1} and v' to strip R_{3i}. Finally, for each vertex u of L_i, add a dummy vertex to R_{3i-1} and a dummy vertex to R_{3i-3}, and connect them to u. The fact that the constructed instance of STRIP PLANARITY is equivalent to the original instance of LEVEL PLANARITY can be proved by observing that, with this construction, each vertex whose degree is greater than 1 has at least a neighbor in both its adjacent strips. We formalize this result in the following.

Theorem 9.3 LEVEL PLANARITY $\propto$ C-PLANARITY *with flat cluster hierarchy consisting of three clusters.*

9.5 Conclusions and Open Problems

In this chapter we show that problem C-SEFE-2 is polynomial-time reducible to problem C-PLANARITY, where the reduction produces instances in which the cluster hierarchy is flat and the underlying graph is a set of paths.

We regard as an intriguing open question whether a polynomial-time reduction exists from general instances of SEFE-2 to C-PLANARITY, which would prove, together with the reduction by Schaefer [Sch13], these two problems to be ultimately the same.

Moreover, as our reduction produces instances of C-PLANARITY with a number of clusters depending linearly on the size of the reduced C-SEFE-2 instance, it is worth of interest asking whether a sublinear or constant number of clusters would suffice. In this direction, we provided a reduction from LEVEL PLANARITY to C-PLANARITY

producing instances with only three clusters. We remark that the complexity of C-PLANARITY is still open even in this setting.

On the other hand, LEVEL PLANARITY, PROPER CL-PLANARITY, and PROPER T-LEVEL PLANARITY are known to be polynomial-time solvable. It would be interesting to understand whether there exist reductions yielding instances that are known to be tractable, e.g., when the instances have only two clusters [BKM98, HN14]. As a final observation, we would like to point out that any reduction from general instances of CL-PLANARITY and T-LEVEL PLANARITY would imply the NP-completeness of C-PLANARITY.

Part V

Drawings with Crossings

Chapter 10

Algorithms and Bounds for Drawing Graphs with Crossing-free Subgraphs

In this chapter[1] we initiate the study of the following problem: *Given a non-planar graph G and a planar subgraph S of G, does there exist a straight-line drawing Γ of G in the plane such that the edges of S are not crossed in Γ by any edge of G?* We give positive and negative results for different kinds of connected spanning subgraphs S of G. Moreover, in order to enlarge the subset of instances that admit a solution, we consider the possibility of bending the edges of G not in S; in this setting we discuss different trade-offs between the number of bends and the required drawing area.

10.1 Introduction

Many papers in graph drawing address the problem of computing drawings of non-planar graphs with the goal of mitigating the negative effect that edge crossings have on the readability of the drawing. Several of these papers describe crossing minimization methods, which are effective and computationally feasible for relatively small and sparse graphs (see [BCG+13] for a survey). Other papers study how non-planar graphs can be drawn such that the "crossing complexity" of the drawing is somewhat controlled, either in the number or in the type of crossings. They include the study of *k-planar drawings*, in which each edge is crossed at most k times (see, e.g., [BEG+12, DDLM13, Did13, EHK+12, HELP12, KM13, PT97]), of *k-quasi pla-*

[1] The contents of this chapter are a joint work with Patrizio Angelini, Carla Binucci, Walter Didimo, Luca Grilli, Fabrizio Montecchiani, Maurizio Patrignani and Ioannis G. Tollis, appeared in [ABD+13] and submitted to journal.

nar drawings, in which no k pairwise crossing edges exist (see, e.g., [Ack09, AT07, GDLM12, FPS13, PSS96, Val98]), and of *large angle crossing drawings*, in which any two crossing edges form a sufficiently large angle (see [DL12] for a survey). Most of these drawings exist only for sparse graphs.

In this chapter we introduce a new graph drawing problem concerned with the drawing of non-planar graphs. Namely: *Given a non-planar graph G and a planar subgraph S of G, decide whether G admits a drawing Γ such that (in Γ) the edges of S are not crossed by any edge of G. Compute Γ if it exists.*

Besides its intrinsic theoretical interest, this problem is also of practical relevance in many application domains. Indeed, distinct groups of edges in a graph may have different semantics, and a group can be more important than another for some applications; in this case a visual interface might attempt to display more important edges without intersections. Furthermore, the user could benefit from a layout in which a connected spanning subgraph is drawn crossing free, since it would support the user to quickly recognize paths between any two vertices, while keeping the other edges of the graph visible.

Please note that the problem of recognizing specific types of subgraphs that are not self-crossing (or that have few crossings) in a given drawing Γ, has been previously studied (see, e.g., [JW93, KSSW07, KLN91, RCUG13]). This problem, which turns out to be NP-hard for most different kinds of instances, is also very different from our problem. Indeed, in our setting the drawing is not the input, but the output of the problem. Also, we require that the given subgraph S is not crossed by any edge of the graph, not only by its own edges.

In this chapter we concentrate on the case in which S is a connected spanning subgraph of G and consider both straight-line and polyline drawings of G. Namely:

(i) In the straight-line drawing setting we prove that if S is any given spanning spider or caterpillar, then a drawing of G where S is crossing free always exists; such a drawing can be computed in linear time and requires polynomial area (Section 10.3), although our construction for caterpillars does not compute integer coordinates. We also show that this positive result cannot be extended to any spanning tree, but we describe a large family of spanning trees that always admit a solution, and we observe that any graph G contains such a spanning tree; unfortunately, our drawing technique for this family of trees may require exponential area. Finally, we characterize the instances $\langle G, S\rangle$ that admit a solution when S is a triconnected spanning subgraph, and we provide a polynomial-time testing and drawing algorithm, whose layouts have polynomial area (Section 10.3).

(ii) We investigate polyline drawings where only the edges of G not in S are allowed to bend. In this setting, we show that all spanning trees can be realized without cross-

ings in a drawing of G of polynomial area, and we describe efficient algorithms that provide different trade-offs between the number of bends per edge and the required drawing area (Section 10.4). Also, we consider the case in which S is any given biconnected spanning subgraph. In this case, we provide a characterization of the positive instances, which yields drawings with polynomial area, if only one bend per edge is allowed.

We finally remark that the study of our problem has been receiving some interest in the Graph Drawing community. In particular, Schaefer proved that given a graph G and a planar subgraph S of G, testing whether there exists a polyline drawing of G where the edges of S are never crossed can be done in polynomial time [Sch14]. This topological variant of the problem has been given the name of PARTIAL PLANARITY and in its definition, differently from our setting, there is no restriction on the number of bends on the edges in $G \setminus S$ and the edges of S are not required to be drawn as straight-line segments. In Chapter 11 we will consider a generalization of the PARTIAL PLANARITY problem, called STREAMED PLANARITY WITH BACKBONE, and show that the PARTIAL PLANARITY problem can be solved in linear time.

In Section 10.2 we give some preliminary definitions that will be used in the rest of the chapter, while in Section 10.5 we discuss conclusions and open problems deriving from this research work.

10.2 Preliminaries and Definitions

Let $G(V, E)$ be a graph and let Γ be a drawing of G in the plane. If all vertices and edge bends of Γ have integer coordinates, then Γ is a *grid drawing* of G, and the *area* of Γ is the area of the minimum bounding box of Γ. We recall that the minimum bounding box of a drawing Γ is the rectangle of minimum area enclosing Γ. If Γ is not on an integer grid, we scale it in order to guarantee the same resolution rule of a grid drawing; namely we expect that the minimum Euclidean distance between any two points on which either vertices or bends of Γ are drawn is at least of one unit. Under this resolution rule, we define the area of the drawing as the area of the minimum bounding box of Γ.

Let $G(V, E)$ be a graph and let $S(V, W)$, $W \subseteq E$, be a spanning subgraph of G. A straight-line drawing Γ of G such that S is crossing-free in Γ (i.e., such that crossings occur only between edges of $E \setminus W$) is called a *straight-line compatible drawing* of $\langle G, S\rangle$. If each edge of $E \setminus W$ has at most k bends in Γ (but still S is drawn straight-line and crossing-free in Γ), Γ is called a *k-bend compatible drawing* of $\langle G, S\rangle$.

If S is a rooted spanning tree of G such that every edge of G not in S connects

either vertices at the same level of S or vertices that are on consecutive levels, then we say that S is a *proper level spanning tree* of G.

A *star* is a tree $T(V, E)$ such that all its vertices but one have degree one, that is, $V = \{u, v_1, v_2, \ldots, v_k\}$ and $E = \{(u, v_1), (u, v_2), \ldots, (u, v_k)\}$; any subdivision of T (including T), is a *spider*: vertex u is the *center* of the spider and each path from u to v_i is a *leg* of the spider. A *caterpillar* is a tree such that removing all its leaves (and their incident edges) results in a path, which is called the *spine* of the caterpillar. The one-degree vertices attached to a spine vertex v are called the *leaves* of v.

In the remainder of the chapter we implicitly assume that G is always a connected graph (if the graph is not connected, our results apply for any connected component).

10.3 Straight-line Drawings

We start studying straight-line compatible drawings of pairs $\langle G, S\rangle$: Section 10.3 concentrates on the case in which S is a spanning tree, while Section 10.3 investigates the case in which S is a spanning triconnected graph.

Spanning Trees

The simplest case is when S is a given Hamiltonian path of G; in this case Γ can be easily computed by drawing all vertices of S in convex position, according to the order they occur in the path. In the following we prove that in fact a straight-line compatible drawing Γ of $\langle G, S\rangle$ can be always constructed in the more general case in which S is a spanning spider (Theorem 10.1), or a spanning caterpillar (Theorem 10.2), or a proper level spanning tree (Theorem 10.3); our construction techniques guarantee polynomial-area drawings for spiders and caterpillars, while requiring exponential area for proper level spanning trees. On the negative side, we show that if S is an arbitrary spanning tree, a straight-line compatible drawing of $\langle G, S\rangle$ may not exist (Lemmas 10.1 and 10.2).

Theorem 10.1 *Let G be a graph with n vertices and m edges, and let S be a spanning spider of G. There exists a grid straight-line compatible drawing Γ of $\langle G, S\rangle$. Drawing Γ can be computed in $O(n + m)$ time and has $O(n^3)$ area.*

Proof: Let u be the center of S and let $\pi_1, \pi_2, \ldots, \pi_k$ be the legs of S. Also, denote by v_i the vertex of degree one of leg π_i $(1 \leq i \leq k)$. Order the vertices of S distinct from u such that: (i) the vertices of each π_i are ordered in the same way they appear in the simple path of S from u to v_i; (ii) the vertices of π_i precede those of π_{i+1} $(1 \leq i \leq k-1)$. If v is the vertex at position j $(0 \leq j \leq n-2)$ in the ordering

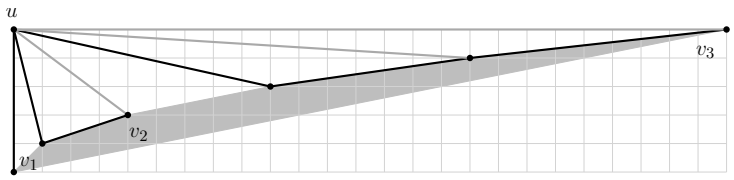


Figure 10.1: Illustration of the drawing construction of Theorem 10.1. The thick edges belong to the spider. The edges of G not incident to u are drawn in the gray convex region.

defined above, draw v at coordinates (j^2, j). Finally, draw u at coordinates $(0, n-2)$. Refer to Fig. 10.1 for an illustration. With this strategy, all vertices of S are in convex position, and they are all visible from u in such a way that no edge incident to u can cross other edges of Γ. Hence, the edges of S do not cross other edges in Γ. The area of Γ is $(n-2)^2 \times (n-2) = O(n^3)$ and Γ is constructed in linear time. □

The following algorithm computes a straight-line compatible drawing of $\langle G, S \rangle$ when S is a spanning caterpillar. Theorem 10.2 proves its correctness, time and area requirements. Although the drawing area is still polynomial, the layout is not a grid drawing.

The basic idea of the algorithm is as follows. It first places the spine vertices of the caterpillar in convex position, along a quarter of circumference. Then, it places the leaf vertices inside the convex polygon formed by the spine vertices, in such a way that they also suitably lie in convex position. With this strategy an edge of the caterpillar will be outside the inner polygon formed by the leaf vertices, and hence it will not cross any edge that connects two leaf vertices. Also, the inner polygon is chosen sufficiently close to the outer polygon formed by the spine vertices in order to guarantee that the edges of the caterpillar are never crossed by other edges incident to the spine vertices (refer to Figs. 10.2 and 10.3 for an illustration). We now formally describe the algorithm.

Algorithm STRAIGHT-LINE-CATERPILLAR. Denote by $u_1, u_2, \ldots, u_k$ the vertices of the spine of S. Also, for each spine vertex u_i $(1 \leq i \leq k)$, let $v_{i1}, \ldots, v_{in_i}$ be its leaves in S (refer to the bottom image in Fig. 10.2(a)). The algorithm temporarily adds to S and G some dummy vertices, which will be removed in the final drawing. Namely, for each u_i, it attaches to u_i two dummy leaves, s_i and t_i. Also, it adds a dummy spine vertex u_{k+1} attached to u_k and a dummy leaf s_{k+1} to u_{k+1} (see the top image in Fig. 10.2(a)). Call G' and S' the new graph and the new caterpillar obtained

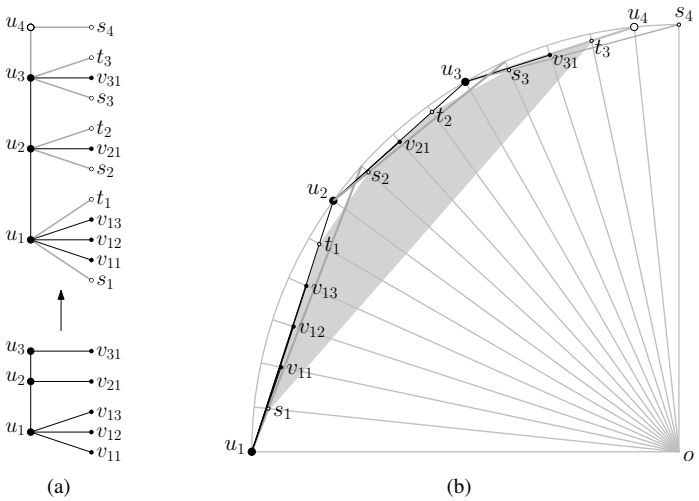


Figure 10.2: Illustration of Algorithm STRAIGHT-LINE-CATERPILLAR: (a) a caterpillar S and its augmented version S'; (b) a drawing of S'; edges of the graph connecting leaves of S are drawn in the gray (convex) region.

by augmenting G and S with these dummy vertices.

The construction of a drawing Γ' of G' is illustrated in Fig. 10.2(b). Consider a quarter of circumference C with center o and radius r. Let N be the total number of vertices of G'. Let $\{p_1, p_2, \ldots, p_N\}$ be N equally spaced points along C in clockwise order, where $\overline{op_1}$ and $\overline{op_N}$ are a horizontal and a vertical segment, respectively. For each $1 \leq i \leq k$, consider the ordered list of vertices $L_i = \{u_i, s_i, v_{i1}, \ldots v_{in_i}, t_i\}$, and let L be the concatenation of all L_i. Also, append to L the vertices u_{k+1} and s_{k+1}, in this order. Clearly the number of vertices in L equals N. For a vertex $v \in L$, denote by $j(v)$ the position of v in L. Vertex u_i is drawn at point $p_{j(u_i)}$ $(1 \leq i \leq k)$; also, vertices u_{k+1} and s_{k+1} are drawn at points p_{N-1} and p_N, respectively. Each leaf v of S' will be suitably drawn along radius $\overline{op_{j(v)}}$ of C; Fig. 10.3 shows a schematic drawing of this construction which has been deliberately deformed to better illustrate its description. More precisely, for any $i \in \{1, \ldots, k\}$, let a_i be the intersection point between segments $\overline{p_{j(u_i)}p_{j(s_{i+1})}}$ and $\overline{op_{j(s_i)}}$, and let b_i be the intersection point between segments $\overline{p_{j(u_i)}p_{j(u_{i+1})}}$ and $\overline{op_{j(t_i)}}$. Vertices s_i and t_i are drawn at points a_i and b_i, respectively. Also, let A_i be the circular arc that is tangent to $\overline{p_{j(u_i)}p_{j(u_{i+1})}}$ at point b_i, and that passes through a_i; vertex v_{ih} is drawn at the intersection point

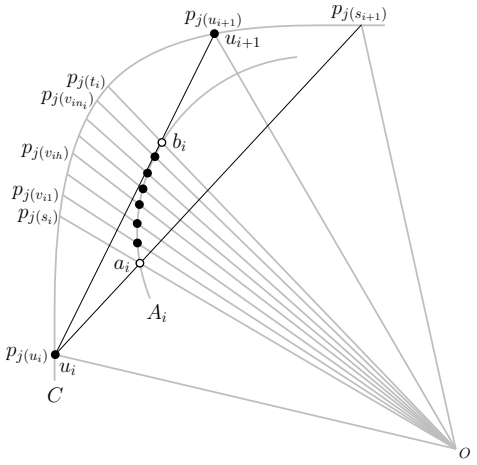


Figure 10.3: Illustration of Algorithm STRAIGHT-LINE-CATERPILLAR: a schematic illustration on how leaves are drawn, the drawing has been deliberately deformed to better visualize its construction.

between A_i and $\overline{op_{j(v_{ih})}}$ $(1 \leq h \leq n_i)$.

Once all vertices of G' are drawn, each edge of G' is drawn in Γ' as a straight-line segment between its end-vertices. Drawing Γ is obtained from Γ' by deleting all dummy vertices and their incident edges.

Theorem 10.2 *Let G be a graph with n vertices and m edges, and let S be a spanning caterpillar of G. There exists a straight-line compatible drawing Γ of $\langle G, S\rangle$. Drawing Γ can be computed in $O(n+m)$ time in the real RAM model and has $O(n^2)$ area.*

Proof: Compute Γ by using Algorithm STRAIGHT-LINE-CATERPILLAR. In the following we first prove that Γ is a straight-line compatible drawing of $\langle G, S\rangle$, and then we analyze time complexity and area requirement. We adopt the same notation used in the description of the algorithm.

CORRECTNESS. We have to prove that in Γ the edges of S are never crossed. For a line ℓ denote by $s(\ell)$ its slope. Our construction places all spine vertices of S' (and hence of S) in convex position. We claim that also all the leaves of S' are in convex position. Indeed, this is clearly true for the subset of leaves of each u_i $(1 \leq i \leq k)$, because this subset is drawn on a circular arc A_i; also, for any $i \in$

$\{1, \ldots, k-1\}$ consider the poly-line connecting the leaves of two consecutive spine vertices u_i, u_{i+1}, in the order they appear in L. This poly-line is convex if and only if the two segments incident to s_{i+1} form an angle λ smaller than π on the side of the plane where the origin o lies. In particular, let ℓ_1 be the line through t_i and s_{i+1}, and let ℓ_2 be the line through s_{i+1} and v, where v coincides with $v_{(i+1)1}$ if such a vertex exists, while v coincides with t_{i+1} otherwise. Angle $\lambda < \pi$ if $s(\ell_1) > s(\ell_2)$. Denote by c the intersection point between the chord $\overline{u_{i+1}u_{i+2}}$ and the radius $\overline{op_{j(v)}}$, and denote by ℓ_3 the line through s_{i+1} and c, we have $s(\ell_3) \geq s(\ell_2)$ by construction; then it suffices to show that $s(\ell_1) > s(\ell_3)$. For any fixed N, $s(\ell_3)$ is maximized when c is as close as possible to s_{i+1}, i.e., when $n_{i+1} = 0$; similarly $s(\ell_1)$ is minimized when t_i is as close as possible to s_{i+1}, i.e., when $n_i = 0$. For $n_i = n_{i+1} = 0$ it can be verified by trigonometry that $\frac{s(\ell_1)}{s(\ell_3)} > 1$ (namely, this ratio tends to 1.23 when N tends to infinity). Hence, the leaves of S' except s_{k+1} form a convex polygon P, which proves the claim.

Now, since by construction the edges of S are all outside P in Γ, these edges cannot be crossed by edges of G connecting two leaves of S. It is also immediate to see that an edge of S cannot be crossed by another edge of S. It remains to prove that an edge of S cannot be crossed by an edge of G connecting either two non-consecutive spine vertices or a leaf of S to a spine vertex of S.

There are two kinds of edges in S. Edges (u_i, u_{i+1}), connecting two consecutive spine vertices, and edges (u_i, v_{ih}), connecting a spine vertex to its leaves. Since by construction Γ is totally drawn inside the closed polygon formed by the spine vertices of S (recall that u_{k+1} and s_{k+1} are dummy vertices, and then they do not belong to Γ), edges (u_i, u_{i+1}) cannot be crossed in Γ. Now, consider an edge (u_i, v_{ih}). Obviously, it cannot be crossed by an edge (u_i, u_j), because two adjacent edges cannot cross in a straight-line drawing; yet, it cannot be crossed by an edge (u_j, u_z) where $j < z$ and $j, z \neq i$. Indeed, if $i < j$ or $i > z$ then there is a line ℓ through o such that (u_i, v_{ih}) completely lies in one of the two half planes determined by ℓ and (u_j, u_z) completely lies in the other half plane; also, if $j < i < z$, then edge (u_i, v_{ih}) completely lies in the open region delimited by (u_j, u_z) and C. We finally show that (u_i, v_{ih}) cannot be crossed by any edge (u_j, v_{df}), where $j \neq i$ and $v_{df} \neq v_{ih}$. Indeed, if $d > i$ and $j > i$, or $d < i$ and $j < i$, then (u_i, v_{ih}) and (u_j, v_{df}) are completely separated by a line through o. Also, if $d < i$ and $j > i$ (or $d > i$ and $j < i$), edge (u_i, v_{ih}) lies completely outside the triangle with vertices o, u_j, v_{df}, thus it cannot cross edge (u_j, v_{df}).

TIME AND AREA REQUIREMENT. It is immediate to see that the construction of Γ' (and then of Γ) can be executed in linear time, in the real RAM model. It remains to prove that the area of Γ is $O(n^2)$. Assume that o coincides with the origin of a Cartesian coordinate system, so that p_1 has coordinates $(-r, 0)$ and p_N has coordi-

nates $(0, r)$. We need to estimate the minimum distance $d_{\min}$ between any two points of Γ. According to our construction, the vertex at position i in L is drawn on a point q_i along radius $\overline{op_i}$, and $d_{\min}$ corresponds to the minimum distance between any two points q_i and q_{i+1} $(1 \leq i \leq N-1)$. Also, denote by p'_i the intersection point between radius $\overline{op_i}$ and the chord $\overline{p_1p_N}$, point q_i is in-between p_i and p'_i along $\overline{op_i}$; this implies that $d'_{\min} \leq d_{\min}$, where $d'_{\min}$ is the minimum distance between any two points p'_i and p'_{i+1}. Now, it is immediate to observe that $d'_{\min}$ equals the length of segment $\overline{p'_{\lceil N/2 \rceil} p'_{\lceil N/2 \rceil + 1}}$. Let p' be the middle point of chord $\overline{p_1p_N}$; p' has coordinates $(\frac{-r}{2}, \frac{r}{2})$. Point p' coincides with $p'_{\lceil N/2 \rceil}$ if N is odd, while it is equidistant to $p'_{\lceil N/2 \rceil}$ and $p'_{\lceil N/2 \rceil + 1}$ if N is even. Hence, denoted by d' the distance between p' and $p'_{\lceil N/2 \rceil + 1}$, we have that $d' \leq d'_{\min} \leq d_{\min}$. Now, let $\alpha = \frac{\pi}{2(N-1)}$ be the angle at o formed by any two radii $\overline{op_i}$, $\overline{op_{i+1}}$, and let β be the angle at o formed by $\overline{op'}$ and $\overline{op'_{\lceil N/2 \rceil + 1}}$. Clearly $\beta = \alpha$ if N is odd, while $\beta = \frac{\alpha}{2}$ if N is even. Also, since $\overline{op'}$ forms a right angle with $\overline{p'p'_{\lceil N/2 \rceil + 1}}$, and since the length of $\overline{op'}$ is $\frac{r}{\sqrt{2}}$, we have that $d' = \frac{r \tan \beta}{\sqrt{2}}$. Hence, if we let $d' = 1$ (which guarantees that $d_{\min} \geq 1$), we obtain $r = \frac{\sqrt{2}}{\tan \beta}$. Since $\tan \beta > \beta$ for $\beta \in (0, \frac{\pi}{2})$, then $r < \frac{\sqrt{2}}{\beta}$, which implies $r = O(N)$, because $\beta = \Theta(\frac{1}{N})$. Thus, the area of Γ is $O(N^2) = O(n^2)$. □

The following lemmas show that, unfortunately, Theorem 10.1 and Theorem 10.2 cannot be extended to every spanning tree S, that is, there exist pairs $\langle G, S \rangle$ that do not admit straight-line compatible drawings, even if S is a ternary or a binary tree.

Lemma 10.1 *Let G be the complete graph on 13 vertices and let S be a complete rooted ternary spanning tree of G. There is no straight-line compatible drawing of $\langle G, S \rangle$.*

Proof: We show by case analysis that there is no straight-line compatible drawing of $\langle G, S \rangle$. Let r be the root of S (see Fig. 10.4(a)). Note that r is the only vertex of S with degree 3. Let u, v, w be the three neighbors of r in S. Two are the cases, either one of u, v, w (say u) lies inside triangle $\triangle(r, v, w)$ (Case 1, see Fig. 10.5(a)), or r lies inside triangle $\triangle(u, v, w)$ (Case 2, see Fig. 10.5(d)).

In Case 1, consider a child u_1 of u. In any straight-line compatible drawing of $\langle G, S \rangle$, u_1 is placed in such a way that u lies inside either triangle $\triangle(u_1, r, w)$ or triangle $\triangle(u_1, r, v)$, assume the former (see Fig. 10.5(b)). Then, consider another child u_2 of u; in order for edge (u, u_2) not to cross any edge, also u_2 has to lie inside $\triangle(u_1, r, w)$, in such a way that both u and u_1 lie inside triangle $\triangle(u_2, r, v)$. This implies that u lies inside $\triangle(u_1, r, u_2)$ (see Fig. 10.5(c)), together with its last child u_3. However, u_3 cannot be placed in any of the three triangles in which $\triangle(u_1, r, u_2)$

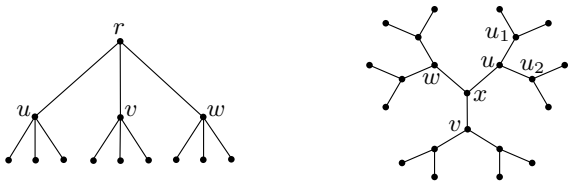


Figure 10.4: (a) The complete rooted ternary tree with 13 vertices in the statement of Lemma 10.1. (b) The complete binary tree with 22 vertices in the statement of Lemma 10.2.

is partitioned by the edges (of S) connecting u to u_1, to r, and to u_2, respectively, without introducing any crossing involving edges of S. This concludes the analysis of Case 1.

In Case 2, note that no child u_1 of u can be drawn inside $\triangle(u, v, w)$, as otherwise one of the edges of S incident to r would be crossed by either $(u, u_1) \in S$, $(u_1, v) \in G$ or $(u_1, w) \in G$. We further distinguish two cases, based on whether u and r lie inside triangle $\triangle(u_1, v, w)$ (Case 2.1, see Fig. 10.5(e)), or r and one of v and w (say w) lies inside triangle $\triangle(u_1, u, v)$ (Case 2.2, see Fig. 10.5(f)). In Case 2.1, consider another child u_2 of u. Note that, u_2 has to lie inside $\triangle(u_1, v, w)$, due to edge $(u, u_2) \in S$. However, u_2 cannot be placed in any of the three regions in which $\triangle(u_1, v, w)$ is partitioned by paths composed of edges of S connecting r to u_1, to v, and to w, respectively. To conclude the proof, note that, if Case 2.2 holds for the children of vertex u, then Case 2.1 must hold for the children of vertex w, as all of them must lie inside $\triangle(u_1, u, v)$. □

Lemma 10.2 *Let G be the complete graph on 22 vertices and let S be a complete unrooted binary spanning tree of G. There is no straight-line compatible drawing of $\langle G, S\rangle$.*

Proof: First, we claim the following property ($P1$): Let x be a vertex of G such that the neighbors u, v, w of x in S are not leaves of S. Then, in any straight-line compatible drawing of $\langle G, S\rangle$, vertex x lies outside triangle $\triangle(u, v, w)$. Observe that property $P1$ directly descends from Case 2 of the proof of Lemma 10.1, where x plays the role of r. Indeed, in that proof, only two of the children of u (and of v and w, symmetrically) were used in the argument.

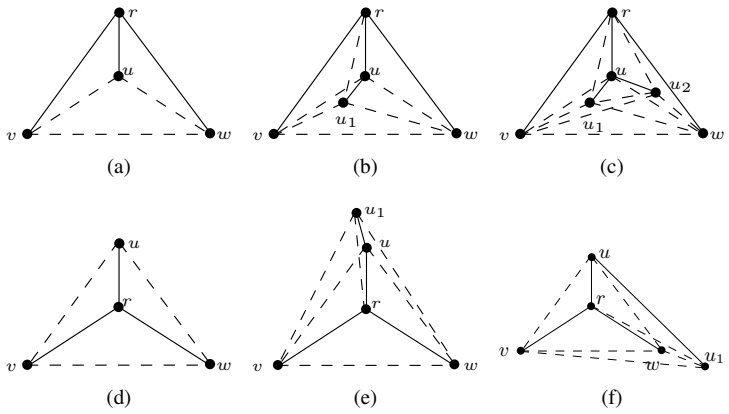


Figure 10.5: Illustration for Lemma 10.1: (a) Case 1 in the proof; u lies inside $\triangle(r, v, w)$. (b) Placement of u_1. (c) Placement of u_2. (d) Case 2 in the proof; r lies inside $\triangle(u, v, w)$. (e) Case 2.1. (f) Case 2.2.

Now, consider the only vertex x of G such that each of the tree subtrees of S rooted at x contains seven vertices (see Fig. 10.4(b)). By $P1$, vertex x lies outside the triangle $\triangle(u, v, w)$ composed of its neighbors u, v, w, which implies that one of u, v, w (say u) lies inside triangle $\triangle(x, v, w)$. As in the proof of Case 1 of Lemma 10.1, with x playing the role of r, we observe that in any straight-line compatible drawing of $\langle G, S\rangle$ in which u lies inside $\triangle(x, v, w)$, the two neighbors u_1 and u_2 of u are placed in such a way that u lies inside $\triangle(u_1, x, u_2)$. While in Lemma 10.1 we used the presence of a fourth neighbor (a third child) of u to prove the statement, here we can apply $P1$, as u_1, x, u_2 are not leaves of S. □

In light of Lemmas 10.1 and 10.2, it is natural to ask whether there are specific subfamilies of spanning trees S (other than paths, spiders, and caterpillars) such that a straight-line compatible drawing of $\langle G, S\rangle$ always exists. The following algorithm gives a positive answer to this question: it computes a straight-line compatible drawing when S is a proper level spanning tree of G. Theorem 10.3 proves the algorithm's correctness, its time complexity, and its area requirement.

The idea of the algorithm is to exploit the properties of proper level spanning trees. Namely, two consecutive levels of a proper level spanning tree induce a subgraph that is a forest of caterpillars and there is no edge spanning more than one level. The algorithm is based on a recursive technique that uses, in each recursive step, an

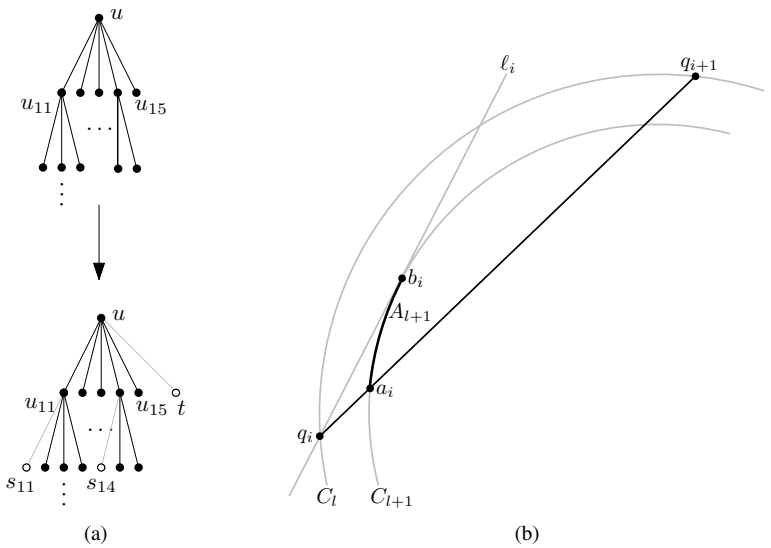


Figure 10.6: Illustration of Algorithm STRAIGHT-LINE-PROPER-LEVEL-SPANNING-TREE: (a) a proper level spanning tree S and its enhanced version S'. (b) Schematic illustration of the inductive step, the curvature of circles C_l and C_{l+1} has been deliberately increased to better visualize the drawing construction.

argument similar to that used for caterpillars, although the geometric construction is different. The final result is a drawing composed of a set of nested convex polygons, one for each level of the proper level spanning tree. Edges connecting vertices of two consecutive levels are drawn between the two corresponding polygons and are not crossed by other edges. All the remaining edges are allowed to cross.

Algorithm STRAIGHT-LINE-PROPER-LEVEL-SPANNING-TREE. Refer to Figs. 10.6 and 10.7 for an illustration of the algorithm. Let u be the root of S (which is at level 0) and let u_{l1},..., u_{lk_l} be the vertices at level $l \in \{1, \ldots, d\}$, where d is the depth of S. The algorithm temporarily adds to S and G some dummy vertices, which will be removed in the final drawing. Namely, for each u_{li}, $1 \leq l \leq d-1$ and $1 \leq i \leq k_l$, it attaches to u_{li} one more (leftmost) child s_{li}. Also, it attaches to root u a dummy (rightmost) child t. Denote by G' and S' the new graph and the new tree, respectively. Notice that S' is still a proper level spanning tree of G'. The algorithm iteratively computes a drawing Γ' of G'. For $l = 1, \ldots, d$, the algorithm defines a circumference

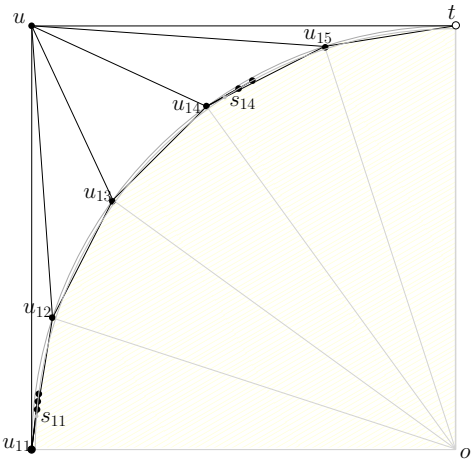


Figure 10.7: Illustration of Algorithm STRAIGHT-LINE-PROPER-LEVEL-SPANNING-TREE: sketch of the final drawing Γ.

C_l with center $o = (0, 0)$ and radius $r_l < r_{l-1}$ ($C_1, \ldots, C_d$ are concentric). The vertices of level l are drawn on the quarter of C_l going from point $(-r_l, 0)$ to point $(0, r_l)$ clockwise. The radius r_{l+1} of the next concentric circumference C_{l+1} is chosen such that C_{l+1} intersects all chords between vertices on C_l.

Let $\{u_{11}, \ldots, u_{1k_1}, t\}$ be the ordered list of the children of root u and let $\{p_{11}, \ldots, p_{1k_1}, p_t\}$ be $k_1 + 1$ equally spaced points along C_1 in clockwise order, where $\overline{op_{11}}$ and $\overline{op_t}$ are a horizontal and a vertical segment, respectively. Vertex u_{1j} is drawn on p_{1j} ($1 \leq j \leq k_1$) and vertex t is drawn on p_t. Also, u is drawn on point $(-r_1, r_1)$.

Assume now that all vertices $u_{l1}, \ldots, u_{lk_l}$ of level l have been drawn ($1 \leq l \leq d-1$) in this order on the sequence of points $\{q_1, \ldots, q_{k_l}\}$, along C_l. The algorithm draws the vertices of level $l+1$ as follows. Let $\overline{q_i q_{i+1}}$ be the chords of C_l, for $1 \leq i \leq k_l - 1$, and let c_l be the shortest of these chords. The radius r_{l+1} of C_{l+1} is chosen arbitrarily in such a way that C_{l+1} intersects c_l in two points and $r_{l+1} < r_l$. This implies that C_{l+1} also intersects every chord $\overline{q_i q_{i+1}}$ in two points. For $1 \leq i \leq k_l$, denote by $L(u_{li}) = \{v_1, \ldots, v_{n_{li}}\}$ the ordered list of children of u_{li} in G'. Also, let a_i be the intersection point between $\overline{q_i q_{i+1}}$ and C_{l+1} that is closest to q_i, and let ℓ_i be the line through q_i tangent to C_{l+1}; denote by b_i the tangent point between ℓ_i and C_{l+1}. Let A_{l+1} be the arc of C_{l+1} between a_i and b_i, and let $\{p_0, p_1, \ldots, p_{n_{li}}\}$ be $n_{li} + 1$ equally spaced points along A_{l+1} in clockwise order. For $v \in L(u_{li})$, denote by $j(v)$ the position of v in $L(u_{li})$. Vertex v_j is drawn on $p_{j(v_j)}$ ($1 \leq j \leq n_{li}$)

and vertex s_{li} is drawn on p_0.

Once all vertices of G' are drawn each edge of G' is drawn in Γ' as a straight-line segment between its end-vertices. Drawing Γ is obtained from Γ' by deleting all dummy vertices and their incident edges.

Theorem 10.3 *Let G be a graph with n vertices and m edges, and let S be a proper level spanning tree of G. There exists a straight-line compatible drawing Γ of $\langle G, S\rangle$. Drawing Γ can be computed in $O(n+m)$ time in the real RAM model.*

Proof: The algorithm that constructs Γ is Algorithm STRAIGHT-LINE-PROPER-LEVEL-SPANNING-TREE. In the following we first prove that Γ is a straight-line compatible drawing of $\langle G, S\rangle$, and then we analyze the time complexity. We adopt the same notation used in the description of the algorithm.

CORRECTNESS. We have to prove that in Γ the edges of S are never crossed. Observe that, since S is a proper level spanning tree of G, there cannot be edges spanning more than two consecutive levels of S. Following its description, we prove the correctness of the algorithm by induction on l.

In the base case consider the levels 0 and 1, also, consider the convex polygon P_1, whose vertices are the points $\{p_1, \ldots, p_{n_u}\}$. All the edges connecting two children of root u are drawn inside P_1, and do not cross the edges connecting the root to its children, which are drawn outside P_1.

Assume by induction that all the edges in S connecting two vertices of two levels $1 \leq l', l'+1 \leq l$ are not crossed in Γ. We prove that all the edges in S connecting two vertices of levels l, $l+1$ are not crossed in Γ. Consider any vertex u_{li} $(1 \leq i \leq k_l)$; observe that all its children are drawn inside the open plane region R defined by the arc of C_l that goes clockwise from q_i to q_{i+1} (where u_{li} and u_{li+1} are drawn) and the chord $\overline{q_i q_{i+1}}$. By construction, R is never intersected by an edge connecting two vertices drawn in a step $l' \leq l$. Consider the convex polygon P_{l+1}, defined by the points where the vertices of level $l+1$ are drawn. All the edges connecting two vertices of level $l+1$ are drawn inside P_{l+1}, and do not cross the edges connecting vertices of level l to their children, which are drawn outside P_{l+1}. It remains to prove that every edge $e' \notin S$, connecting a vertex of level l to a vertex of level $l+1$, does not cross any edge $e \in S$, connecting a vertex of level l to a vertex of level $l+1$. In particular, let $e = (u_{li}, u_{l+1j}) \in S$ $(1 \leq i \leq k_l$ and $1 \leq j \leq k_{l+1})$ and $e' = (u_{lz}, u_{l+1f}) \notin S$ $(1 \leq z \leq k_l$ and $1 \leq f \leq k_{l+1})$. Assume that u_{l+1f} is not a child of u_{li} in S. If $i < z$ and $j < f$ or $i > z$ and $j > f$, then there is a line ℓ through o such that e completely lies in one of the two half planes determined by ℓ and e' completely lies in the other half plane. If $i < z$ and $j > f$ or $i > z$ and $j < f$, consider the line ℓ containing the straight-line segment $\overline{u_{li}u_{lz}}$, then e completely lies in one of the two half planes

determined by ℓ (the one containing u_{l+1j}), and e' completely lies in the other half plane; indeed, r_{l+1} has been chosen so that it intersects c_l (the minimum-length chord of C_l). Finally, suppose u_{l+1f} is a child of u_{li} in S; by construction, any edge that connects u_{lz} to a vertex of level $l+1$ (which is not a child of u_{lz}), including e', must cross the circumference C_{l+1} exactly once (near the point where u_{lz} is placed).

TIME REQUIREMENT. At each inductive step, the technique performs a number of operations proportional to $k_l + k_{l+1}$. Indeed, c_l is chosen by looking only at the chords between consecutive points on C_l. Hence, the overall time complexity is $O(\sum_{l=0}^{d-1}(k_l + k_{l+1}) + m) = O(\sum_{l=0}^{d-1} k_l) + O(\sum_{l=0}^{d-1} k_{l+1}) + O(m) = O(n+m)$.

AREA REQUIREMENT. Observe that the compatible drawing computed by Algorithm STRAIGHT-LINE-PROPER-LEVEL-SPANNING-TREE may require area $\Omega(2^n)$. Indeed, let $\mathcal{L}(C_1)$ be the length of C_1; the children of the root u are drawn along an arc A_1 of C_1 whose length is $\mathcal{L}(A_1) < \mathcal{L}(C_1)/2$. Inductively, the children of any vertex of level $l-1$ are drawn along an arc A_l of C_l whose length is $\mathcal{L}(A_l) < \mathcal{L}(A_{l-1})/2 < \mathcal{L}(C_1)/2^l$. Hence, the children of any vertex of level $d-1$ are drawn along an arc of circumference A_d whose length is $\mathcal{L}(A_d) < \mathcal{L}(C_1)/2^d$. It follows that the minimum distance between any two points in Γ is $d_{\min} = o(\mathcal{L}(C_1)/2^d)$. Consider the case $d \in O(n)$, and impose $d_{\min} = 1$, it follows that $\mathcal{L}(C_1) \in \Omega(2^n)$, which implies that the area of Γ is $\Omega(2^n)$.

As an upper bound, we prove that the area requirement of our algorithm is $2^{O(n)}$. Consider again the arc A_l, where the children of any vertex of level $l-1$ will be drawn. Namely, denote by q_i the point on C_{l-1} where a vertex v of level $l-1$ has been placed, and by q_{i+1} its consecutive point on C_{l-1}. Recall that the circumference C_l is chosen such that it intersects the chord $s_{l-1} = \overline{q_i q_{i-1}}$ in two distinct points, denoted by a_i and a'_i. We can choose C_l such that the distance between a_i and q_i is the same as the distance between a'_i and q_{i+1}, and such that the length of $s_l = \overline{a_i a'_i}$ is $\mathcal{L}(s_l) = \frac{\mathcal{L}(s_{l-1})}{2}$. To this end, recall that $\mathcal{L}(s_{l-1}) = 2r_{l-1}\sin\beta_{l-1}$, where β_{l-1} is half of the central angle defined by the chord s_{l-1}. We choose a radius r_l for C_l such that $r_l = \frac{r_{l-1}}{2}\frac{\sin\beta_{l-1}}{\sin\frac{\beta_{l-1}}{2}}$. The symmetry of this choice implies that $\mathcal{L}(A_l) > \frac{\mathcal{L}(s_l)}{2} = \frac{\mathcal{L}(s_{l-1})}{4}$. Denote by u the parent node of v in S' and let k_u be the number of children of u in S'. Since the children of u (including v) are placed equispaced along A_{l-1}, we have that $\mathcal{L}(s_{l-1}) = \frac{\mathcal{L}(A_{l-1})}{k_u}$. Thus, $\mathcal{L}(A_l) > \frac{1}{4}\frac{\mathcal{L}(A_{l-1})}{k_u}$. Inductively, we obtain for a tree of depth d, $\mathcal{L}(A_d) > \frac{1}{4^d}\frac{\mathcal{L}(A_1)}{\prod_{\forall u} k_u}$. Since $\sum_{\forall u} k_u < n'$ we have $\prod_{\forall u} k_u < 2^{n'}$. Imposing the minimum distance $d_{min} < \mathcal{L}(A_d)$ equal to 1, we obtain $\mathcal{L}(C_1) = \mathcal{L}(A_1) < 4^d 2^{n'}$. If $d = O(n')$ and $n' = O(n)$, we obtain that the area of the drawing is $2^{O(n)}$. □

It is worth observing that any graph G admits a proper level spanning tree rooted at an arbitrarily chosen vertex r of G. Indeed, it corresponds to the spanning tree computed with a breadth-first-search starting from r. Thus, each graph admits a straight-line drawing Γ such that one of its spanning trees S is never crossed in Γ.

Spanning Triconnected Subgraphs

Here we focus on the case in which S is a triconnected spanning subgraph of G. Clearly, since every tree can be augmented with edges to become a triconnected graph, Lemmas 10.1 and 10.2 imply that, if S is a triconnected graph, a straight-line compatible drawing of $\langle G, S\rangle$ may not exist. The following lemma characterizes those instances for which such a drawing exists.

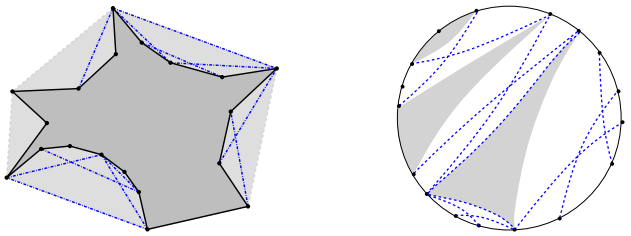
Lemma 10.3 *Let $G(V, E)$ be a graph, $S(V, W)$ be a planar triconnected spanning subgraph of G, and $\mathcal{E}$ be the unique planar (combinatorial) embedding of S (up to a flip). A straight-line compatible drawing Γ of $\langle G, S\rangle$ exists if and only if:*

(1) Each edge $e \in E \setminus W$ connects two vertices belonging to the same face of $\mathcal{E}$.

(2) There exists a face f of $\mathcal{E}$ containing three vertices u, v, w such that, for any edge $(x, y) \in E \setminus W$ with $x, y \in f$, vertices x and y are not separated by u, v, w, that is, vertices u, v, w appear on the same path between x and y along the boundary of f.

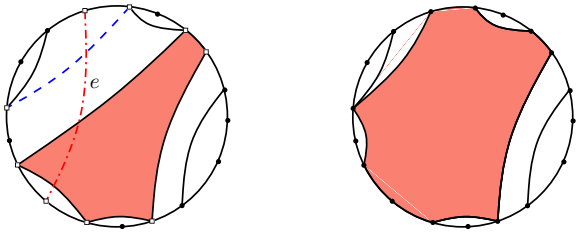
Proof: First we prove the necessity of the conditions. Suppose that $\langle G, S\rangle$ admits a straight-line compatible drawing Γ. Condition 1 is trivially satisfied. In fact, for each edge $e \in E \setminus W$, the segment representing e in Γ must be drawn inside a face of $\mathcal{E}$, as otherwise a crossing between e and some edge of S would occur. Regarding Condition 2, consider the circular sequence $v_1, v_2, \ldots, v_k$ of vertices of V on the convex hull of Γ (see Fig. 10.8(a)). Observe that $k \geq 3$ and that any triple of such vertices satisfies Condition 2.

Then we prove the sufficiency of the conditions. Suppose that there exist three vertices v_1, v_2, and v_3 of a face f of $\mathcal{E}$ satisfying Condition 2 (see Fig. 10.8(b)). Consider the graphs $G^*(V, E \cup \Delta)$ and $S^*(V, W \cup \Delta)$, where $\Delta = \{(v_1, v_2), (v_2, v_3), (v_1, v_3)\}$. Observe that, due to Condition 2, S^* is a triconnected planar spanning subgraph of G^* and that Δ forms an empty triangular face in the unique planar embedding of S^*. Produce a strictly-convex drawing Γ^* of S^* with Δ as the external face (for example, using the algorithm in [BR06]). A straight-line compatible drawing Γ of $\langle G, S\rangle$ can be obtained from Γ^* by removing the edges in Δ and by adding the edges of $E \setminus W$.



(a) (b)

Figure 10.8: (a) A straight-line compatible drawing of $\langle G, S\rangle$ used to show the necessity of Condition 2. (b) A face of $\mathcal{E}$, where edges of $E \setminus W$ are drawn as dashed curves. Shaded triangles identify three triplets of vertices among those satisfying Condition 2.



(a) (b)

Figure 10.9: Two consecutive steps of Algorithm STRAIGHT-LINE-TRICONNECTED-DECISION. (a) The outerplane graph G_f; the shaded face is `full` (the others are `empty`); the dashed-dotted edge e is the next edge of E_f to be considered; edges in E_χ are drawn as dashed lines; white squares are vertices of V_χ. (b) Graph G_f after the update due to edge e.

Observe that by Condition 1 and by the strict convexity of the faces of Γ^*, the edges of $E \setminus W$ do not intersect edges of S. □

In the following we describe an algorithm to test in polynomial time whether the conditions of Lemma 10.3 are satisfied by a pair $\langle G, S\rangle$ in which S is triconnected and spanning.

Algorithm STRAIGHT-LINE-TRICONNECTED-DECISION. Let $\mathcal{E}$ be the unique planar embedding of S (up to a flip). The algorithm verifies that each edge of $E \setminus W$

satisfies Condition 1 of Lemma 10.3 and that there exists a face f of $\mathcal{E}$ containing three vertices v_1, v_2, and v_3, that satisfy Condition 2 of Lemma 10.3. If both conditions hold, then v_1, v_2, and v_3 can be used to find a straight-line compatible drawing Γ of $\langle G, S\rangle$ as described in the proof of Lemma 10.3.

Condition 1 is can be tested by checking, for each edge e of W, whether the endpoints of e have distance 2 in the vertex-face incidence graph of S by using the algorithm described in [KK03] that tests the existence of constant-length paths between vertices of a planar graph.

To test Condition 2 of Lemma 10.3 we perform the following procedure on each face f of $\mathcal{E}$, restricting our attention to the set E_f of edges in $E \setminus W$ whose both end-vertices belong to f. We maintain an auxiliary outerplane graph G_f whose vertices are the vertices V_f of f. Each internal face of G_f is either marked as `full` or as `empty`. Faces marked `full` are not adjacent to each other. Intuitively, we have that any three vertices of an `empty` face satisfy Condition 2, while no triple of vertices of a `full` face satisfies such a condition. We initialize G_f with the cycle composed of the vertices and the edges of f and mark its unique internal face as `empty`. At each step an edge e of E_f is considered and G_f is updated accordingly. If the end-vertices of e belong to a single `empty` face of G_f, then we update G_f by splitting such a face into two `empty` faces. If the end-vertices of e belong to a single `full` face, then we ignore e, as adding e to G_f would determine crossings between e and several edges and faces (see Fig. 10.9(a)). Consider the set E_χ of internal edges of G_f crossed by e. Define a set of vertices V_χ of G_f containing the end-vertices of e, the end-vertices of edges of E_χ that are incident to two `empty` faces, and the vertices of the `full` faces traversed by e. Remove all edges in E_χ from G_f. Mark the face f' obtained by such a removal as `empty`. Form a new face f_χ inside f' with all vertices in V_χ by connecting them as they appear in the circular order of f, and mark f_χ as `full` (see Fig. 10.9(b)).

When all the edges of E_f have been considered, if G_f has an internal face marked as `empty`, any three vertices of this face satisfy Condition 2. Otherwise, G_f has a single internal face marked `full` and no triple of vertices of f satisfies Condition 2.

Theorem 10.4 *Let $G(V, E)$ be a graph and let $S(V, W)$ be a planar triconnected spanning subgraph of G. There exists an $O(|V|\times|E\setminus W|)$-time algorithm that decides whether $\langle G, S\rangle$ admits a straight-line compatible drawing Γ and, in the positive case, computes it on an $O(|V|^2) \times O(|V|^2)$ grid.*

Proof: First, apply Algorithm STRAIGHT-LINE-TRICONNECTED-DECISION to decide whether $\langle G, S\rangle$ satisfies the two conditions of Lemma 10.3. If this is the case,

apply the algorithm described in the proof of Lemma 10.3 to construct a straight-line compatible drawing Γ of $\langle G, S \rangle$.

CORRECTNESS. We prove that the STRAIGHT-LINE-TRICONNECTED-DECISION algorithm correctly checks Conditions 1 and 2 of Lemma 10.3.

Condition 1 is easy to check, since the vertex-face incidence graph of a planar graph is trivially planar and hence the algorithm described in [KK03] can be used to correctly test whether two vertices have distance 2 in such graph.

Regarding Condition 2, suppose that the algorithm identifies a face f of $\mathcal{E}$ such that a face f_e marked `empty` can be found in the auxiliary outerplane graph G_f. Since f_e is not traversed by any edge of E_f, we have that any three vertices of f_e satisfy Condition 2 of Lemma 10.3.

Conversely, suppose that the computation for every face f of $\mathcal{E}$ yields a single internal face of the auxiliary outerplane graph G_f marked `full`. Then we prove that Condition 2 of Lemma 10.3 is not satisfied. Our proof is based on the invariant that during the computation of Algorithm STRAIGHT-LINE-TRICONNECTED-DECISION, as well as at its end, any face of G_f marked `full` cannot contain three vertices that satisfy Condition 2 of Lemma 10.3. In order to see this, first observe that a pair of vertices that separates the end-vertices of some edge in the circular order of a face f of $\mathcal{E}$ is also separated by such an edge. Hence, searching for three vertices of f that do not separate the end-vertices of any edge of E_f is equivalent to searching for three vertices that are not separated by any edge in E_f. Therefore, it suffices to prove the following statement: any pair of vertices of a `full` face f_{full} that is non-contiguous in the circular order of f_{full} is separated by an edge of E_f. In fact, since a `full` face has at least four vertices, this implies that Condition 2 of Lemma 10.3 cannot be satisfied by any three vertices of f_{full}.

First, we prove inductively on the size of a `full` face f_{full} that any edge that crosses the boundary of f_{full} also crosses one edge of E_f. In the base case, when f_{full} is created, the current edge e of E_f crosses a set of edges E_χ separating `empty` faces (i.e., no previous `full` face involved). In this case, any vertex of V_χ is the end-vertex of an edge in E_χ or of e, implying the statement. In the general case, when f_{full} is created, the current edge e of E_f crosses both a set of edges E_χ separating `empty` faces and a set of `full` faces of G_f. Applying an inductive argument it can be proved that all vertices of V_χ are the end-vertices of some edge of E_f, thus proving that e crosses at least one edge of E_f.

We are now ready to prove inductively that any pair of vertices of a `full` face f_{full} that is non-contiguous in the circular order of f_{full} is separated by an edge of E_f. As above, in the base case (when face f_{full} is created) the current edge $e = (v_1, v_2)$ of E_f does not cross any `full` face. Since edges in E_χ do not cross each other, it

is easy to verify that any pair of vertices of V_χ that are non-contiguous in the circular order of f_{full} are separated either by an edge in E_χ or by e. In the inductive case, when face f_{full} is created the current edge e of E_f crosses both a set of edges E_χ separating `empty` faces and a set of `full` faces of G_f. Consider a pair of vertices u, v of V_χ that are non-contiguous in the circular order of f_{full}. Observe that, if u and v are separated by some edge, adding edge (u, v) would introduce a crossing. If both u and v belong to the same `full` face that is merged into f_{full}, then by applying an inductive argument we have that they are separated by an edge of E_f. Otherwise, either edge (u, v) would cross the boundary of at least one `full` face or it would cross e. In both cases edge (u, v) would cross an edge of E_f, proving that u and v are separated by some edge of E_f.

TIME AND AREA REQUIREMENTS. The unique planar embedding $\mathcal{E}$ of S (up to a flip) can be computed in $O(|V|)$ time. Regarding the time complexity of testing Condition 1, the vertex-face incidence graph I of S can be constructed in time linear in the size of S. Since I is a planar graph, deciding if two vertices have distance 2 in such graph can be done in constant time provided that an $O(|V|)$-time preprocessing is performed [KK03]. Thus, testing Condition 1 for all edges in $E \setminus W$ can be done in $O(|V| + |E \setminus W|)$ time. Regarding the time complexity of testing Condition 2, observe that, for each face f of $\mathcal{E}$, E_f can be computed in $O(|V| + |E \setminus W|)$ time while verifying Condition 1. Since for each face f of $\mathcal{E}$, the size of G_f is $O(|V_f|)$, adding edges in E_f has time complexity $O(|E_f| \times |V_f|)$. Overall, as $\sum_{f \in \mathcal{E}} |E_f| = O(|E \setminus W|)$ and $\sum_{f \in \mathcal{E}} |V_f| = O(|V|)$, the time complexity of testing Condition 2 is $O(|E \setminus W| \times |V|)$, which gives the time complexity of the algorithm. Regarding the area, the algorithm in [BR06] can be used to obtain in linear time a strictly-convex grid drawing of S^* on an $O(|V|^2) \times O(|V|^2)$ grid. □

10.4 Polyline Drawings

In this section we allow bends along the edges of G not in S, while we still require that the edges of S are drawn as straight-line segments. Of course, since edge bends are negatively correlated to the readability of the drawing, our goal is to compute k-bend compatible drawings for small values of k. However, it might happen that the number of bends in the drawing can only be reduced at the cost of increasing the required area. Throughout the section, we discuss possible trade-offs between these two measures of the quality of the drawing.

We split the section into two subsections, dealing with the case in which S is a spanning tree and with the case in which S is a biconnected spanning graph, respectively.

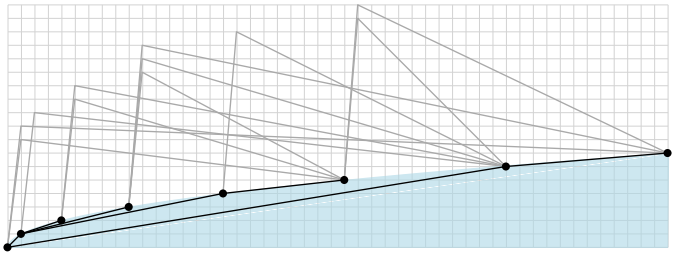


Figure 10.10: Illustration of Algorithm ONE-BEND TREE.

Spanning Trees

In this subsection we prove that allowing bends along the edges of G not in S permits us to compute compatible drawings of pairs $\langle G, S\rangle$ for every spanning tree S of G; such drawings are realized on a polynomial-area integer grid. We provide algorithms that offer different trade-offs between number of bends and drawing area.

Let $G(V, E)$ be a graph with n vertices and m edges, and let $S(V, W)$ be any spanning tree of G. We denote by $x(v)$ and $y(v)$ the x- and the y-coordinate of a vertex v, respectively. The following algorithm computes a 1-bend compatible drawing of $\langle G, S\rangle$.

Algorithm ONE-BEND TREE. The algorithm works in two steps (refer to Fig. 10.10).

STEP 1: Consider a point set of size n such that for each point p_i, the x- and y-coordinates of p_i are i^2 and i, respectively. Construct a straight-line drawing of S by placing the vertices on points p_i, $1 \leq i \leq n$, according to a DFS traversal.

STEP 2: Let v_i be the vertex placed on point p_i. For each $i \in \{1, \ldots, n\}$, draw each edge $(v_i, v_j) \in E \setminus W$ such that $j > i$ as a polyline connecting p_i and p_j, and bending at point $(i^2 + 1, n + c)$ where c is a progressive counter, initially set to one.

Theorem 10.5 *Let $G(V, E)$ be a graph with n vertices and m edges, and let $S(V, W)$ be any spanning tree of G. There exists a 1-bend compatible drawing Γ of $\langle G, S\rangle$. Drawing Γ can be computed in $O(n + m)$ time and has $O(n^2(n + m))$ area.*

Proof: The algorithm that constructs Γ is Algorithm ONE-BEND TREE. In the following we first prove that Γ is a straight-line compatible drawing of $\langle G, S\rangle$, and then we analyze the time and area complexity. We adopt the same notation used in the description of the algorithm.

CORRECTNESS. We have to prove that Γ satisfies the following properties: (i) the edges of S are never crossed and (ii) there exists no overlapping between a bend-point and an edge in $E \setminus W$. To prove (i), observe that the drawing of S contained in Γ is planar and that the edges in $E \setminus W$ are drawn outside the convex region containing the drawing of S. To prove (ii), observe that, for each two edges (v_i, v_j) and (v_p, v_q) such that $i < j$, $p < q$, and $p < i$, the bend-point of the polyline representing (v_i, v_j) lies above the polyline representing (v_p, v_q).

TIME AND AREA REQUIREMENT. Concerning time complexity, STEP 1 can be performed in $O(n)$ time. STEP 2 can be performed in $O(m)$ time, since for each edge in $E \setminus W$ a constant number of operations is required. Concerning area requirements, the width of Γ is $O(n^2)$, by construction, while the height of Γ is given by the y-coordinate of the topmost bend-point, that is $n + m$.

□

Next, we describe an algorithm that constructs 3-bend compatible drawings of pairs $\langle G, S \rangle$ with better area bounds than Algorithm ONE-BEND TREE for sparse graphs. This algorithm also produces drawings with optimal crossing angular resolution, i.e., edges cross only at right angles. Drawings of this type, called *RAC drawings*, have been widely studied in the literature [DEL11, DL12] and are motivated by cognitive studies showing that drawings where the edges cross at very large angles do not affect too much the readability of the drawing [HHE08].

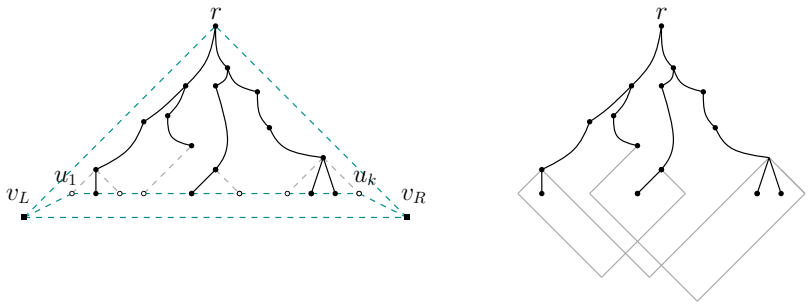
Algorithm THREE-BEND TREE. The algorithm works in four steps (refer to Fig. 10.11(b) for an illustration).

STEP 1: Let G' be the graph obtained from G by subdividing each edge $(v_i, v_j) \in E \setminus W$ with two dummy vertices $d_{i,j}$ and $d_{j,i}$. Let S' be the spanning tree of G', rooted at any non-dummy vertex r, obtained by deleting all edges connecting two dummy vertices. Clearly, every dummy vertex is a leaf of S'.

STEP 2: For each vertex of S', order its children arbitrarily, thus inducing a left-to-right order of the leaves of S'. Rename the leaves of S' as $u_1, \ldots, u_k$ following this order. For each $i \in \{1, \ldots, k-1\}$, add an edge (u_i, u_{i+1}) to S'. Also, add to S' two dummy vertices v_L and v_R, and edges (v_L, r),(v_R, r),(v_L, u_1),(u_k, v_R), (v_L, v_R).

STEP 3: Construct a straight-line grid drawing Γ' of S', as described in [Kan96], in which edge (v_L, v_R) is drawn as a horizontal segment on the outer face, vertices $u_1, \ldots, u_k$ all lie on points having the same y-coordinate Y, and the rest of S' is drawn above such points. Remove from Γ' the vertices and edges added in STEP 2.

STEP 4: Compute a drawing Γ of G such that each edge in W is drawn as in Γ', while each edge $(v_i, v_j) \in E \setminus W$ is drawn as a polyline connecting v_i and v_j, bending at



(a) (b)

Figure 10.11: Illustration of Algorithm THREE-BEND TREE: (a) Drawing Γ' produced at STEP 3 in which the vertices and the edges added in STEP 2 are represented by dashed segments and (b) drawing Γ produced at STEP 4. Edges of S are represented by curve segments to indicate that the drawing of S in Γ' depends on the layout produced by the algorithm in [Kan96]

$d_{i,j}$, at $d_{j,i}$, and at a point $(a, Y-b)$ where $a = \frac{x(d_{i,j})+x(d_{j,i})}{2}$ and $b = \frac{|x(d_{i,j})-x(d_{j,i})|}{2}$. Finally, scale drawing Γ by a factor of 2.

Theorem 10.6 *Let $G(V, E)$ be a graph with n vertices and m edges, and let $S(V, W)$ be any spanning tree of G. There exists a grid 3-bend compatible drawing Γ of G, which is also a RAC drawing. Drawing Γ can be computed in $O(n+m)$ time and has $O((n+m)^2)$ area.*

Proof: The algorithm that constructs Γ is Algorithm THREE-BEND TREE. In the following we first prove that Γ is a straight-line compatible drawing of $\langle G, S\rangle$, and then we analyze the time and area complexity. We adopt the same notation used in the description of the algorithm.

CORRECTNESS. We have to prove that Γ satisfies the following properties: (i) the edges of S are never crossed and (ii) there exists no overlapping between a bend-point and an edge in $E \setminus W$. To prove (i), observe that the drawing of S contained in Γ is planar [Kan96] and lies above the horizontal line with y-coordinate Y. Also, observe that each edge-segment that is drawn by STEP 3 and STEP 4 either lies below such line or has the same representation as an edge of S' in Γ'. To prove (ii), observe that, for each edge e in $E \setminus W$, the first and the last bend-points have the same position as dummy vertices of S' in Γ', and the part of e between such two endpoints is composed

of two segments whose slopes are $\frac{\pi}{4}$ and $-\frac{\pi}{4}$. Finally, it is immediate to see that each edge of W has at most three bends, and that two edge segments can only intersect if they have slopes equal to $\frac{\pi}{4}$ or to $-\frac{\pi}{4}$, thus Γ is a RAC compatible drawing of $\langle G, S\rangle$.

TIME AND AREA REQUIREMENT. Concerning time complexity, STEP 1 can clearly be performed in $O(m)$ time; STEP 2 in $O(n+m)$ time; STEP 3 in $O(n+m)$ time [Kan96]; and STEP 4 in $O(n+m)$ time, since for each edge in E a constant number of operations is required. Concerning area requirements, by construction and by the requirements of the algorithm in [Kan96], both the width and the height of Γ are $O(n+m)$. Note that, by construction, each vertex lies on a grid point in Γ; also, scaling Γ by a factor of 2 ensures that each bend lies on a grid point, as well.

□

Biconnected Spanning Subgraphs

In this subsection we consider the case in which $S(V, W)$ is a biconnected spanning subgraph. We note that, in this case, an example of a negative instance $\langle G, S\rangle$ can be directly inherited from the case in which S is a triconnected graph (Section 10.3), as any triconnected graph is also a biconnected graph. More generally, a trivial necessary condition for an instance to be positive is that S admits an embedding in which for every edge e of $E \setminus W$ the two end-vertices of e share a face. In the following we prove that this condition is also sufficient if the edges of $E \setminus W$ are allowed to bend, hence providing a characterization of the positive instances. Note that, we are able to produce compatible drawings of such instances with polynomial area in which each edge bends at most once. Furthermore, we describe how to test the existence of an embedding satisfying the characterization in linear time.

We start by showing an algorithm that, given an instance $\langle G, S\rangle$ that admits an embedding in which for every edge e of $E \setminus W$ the two end-vertices of e share a face, produces a 1-bend compatible drawing of $\langle G, S\rangle$ with polynomial area.

Algorithm 1-BEND-BICONNECTED-DRAWING. Let $\mathcal{E}$ be a planar embedding of S such that each edge $e \in E \setminus W$ is incident to vertices belonging to the same face of $\mathcal{E}$.

For each face f of $\mathcal{E}$, add to S a dummy vertex v_f inside f and connect it to each vertex of f, hence obtaining a planar graph S' with $|S'| = O(|V|)$. Produce any straight-line planar drawing Γ' of S' in $O(|S'|\times|S'|) = O(|V|\times|V|)$ area by applying any of the known algorithms, say the one by De Fraissex, Pach, and Pollack [dPP90]. Then, obtain a straight-line planar drawing Γ^* of S in $O(|V| \times |V|)$ area by removing from Γ' all the dummy vertices and their incident edges. Observe that, Γ^* is a *star-shaped* drawing, namely each face is represented by a polygon whose kernel is not empty; in particular, since the area of Γ^* is $O(|V| \times |V|)$, for each face f there exists

a disk of radius $O(\frac{1}{|V|})$ centered at a grid point completely lying inside the kernel of the polygon representing f. This is due to the fact that (i) the grid point where vertex v_f used to be placed in Γ' belongs to the kernel of the polygon representing f in Γ^* and that (ii) for any three grid points p, p_1, p_2 of a grid of size $|V| \times |V|$, the slopes of segments $\overline{pp_1}$ and $\overline{pp_2}$ differ by at least $\frac{1}{|V|}$.

Further, scale Γ^* by a factor of $|V| \cdot |E \setminus W| + 1$ along the x-axis (where the additive factor 1 is needed to handle the degenerate case in which $|E \setminus W| = 0$). This implies that for each face f the kernel of the polygon representing f in Γ^* contains an ellipse centered at a grid point whose vertical radius is at least $\frac{1}{|V|}$ and whose horizontal radius is at least $|E \setminus W| + 1$. Note that, this ellipse contains at least $|E \setminus W| + 1$ grid points in its interior.

Finally, obtain Γ from Γ^* as follows. For each edge e of $E \setminus W$, let f_e be any face of Γ^* containing both the end-vertices of e. Draw e as a polyline whose unique bend-point is placed on a point inside the kernel of the polygon representing f_e in Γ^* in such a way that there exists no overlapping between a bend-point and an edge.

Lemma 10.4 *Let $G(V, E)$ be a graph and let $S(V, W)$ be a spanning biconnected planar subgraph of G. Then, $\langle G, S \rangle$ admits a 1-bend compatible drawing Γ with area $O((|V|^2 \cdot |E \setminus W| + |V|) \times |V|)$ if and only if there exists a planar (combinatorial) embedding $\mathcal{E}$ of S such that each edge $e \in E \setminus W$ connects two vertices belonging to the same face of $\mathcal{E}$.*

Proof: We prove the necessity. Suppose that $\langle G, S \rangle$ admits a 1-bend compatible drawing Γ. Consider the drawing Γ' of S contained in Γ and let $\mathcal{E}$ be the planar embedding of S determined by Γ'. For each edge $e \in E \setminus W$, the polyline representing e in Γ must be drawn inside a face of $\mathcal{E}$, as otherwise a crossing between e and some edge of S would occur.

We prove the sufficiency. Suppose that there exists an embedding $\mathcal{E}$ of S in which for every edge e of $E \setminus W$ the two end-vertices of e share a face. Then, apply Algorithm 1-BEND-BICONNECTED-DRAWING to construct a 1-bend compatible drawing Γ of $\langle G, S \rangle$ with area $O((|V|^2 \cdot |E \setminus W| + |V|) \times |V|)$ in which the embedding of S is $\mathcal{E}$. We prove that Algorithm 1-BEND-BICONNECTED-DRAWING correctly computes such a drawing.

CORRECTNESS. We prove that Algorithm 1-BEND-BICONNECTED-DRAWING correctly constructs a 1-bend compatible drawing Γ of $\langle G, S \rangle$. Indeed, the addition of a dummy vertex inside every face of the computed embedding of S can always be performed while maintaining the property that the resulting graph is planar and simple, which implies that a planar star-shaped drawing of S preserving $\mathcal{E}$ can be constructed.

As observed above, the kernel of the polygon representing each face of $\mathcal{E}$ in such a drawing contains at least $|E \setminus W| + 1$ grid points; hence, non-overlapping bend-points for all the edges lying in a face can be placed inside the kernel of this face.

TIME AND AREA REQUIREMENTS. A star-shaped drawing of S can be found in linear time by applying the algorithm in [dPP90], as only a linear number of dummy vertices is added to S. Also, scaling the obtained drawing can be done in linear time. Finally, each edge of $E \setminus W$ can be routed inside a face of $\mathcal{E}$ in constant time. The area bound derives from the fact that (i) drawing Γ' obtained by applying the algorithm in [dPP90] has area $|V| \times |V|$, that (ii) Γ^* has the same area as Γ', and that (iii) Γ is obtained by scaling Γ^* by a factor of $|V| \cdot |E \setminus W| + 1$ along the x-axis. □

In Lemma 10.4 we presented a necessary and sufficient condition for an instance $\langle G, S \rangle$ in which S is biconnected and spanning to admit a 1-bend compatible drawing with polynomial area. In the following we describe an algorithm that tests in polynomial time whether such a condition is satisfied by $\langle G, S \rangle$ and, if this is the case, returns an embedding $\mathcal{E}$ of S with the properties required by Lemma 10.4.

Our algorithm exploits a reduction to instances of SUNFLOWER SEFE. Refer to [BKR13b, Chapter 11], to Section 3.2, and to Chapter8 for discussions on this problem. Recall that, the SUNFLOWER SEFE problem takes as input a set of k planar graphs $\langle G_i(V, E_i) \rangle_{i=1}^{k}$ on the same set V of vertices such that, for each two graphs G_i and G_j, with $i \neq j$, $G_i \cap G_j = G_\cap$, where $G_\cap = \bigcap_{l=1}^{k} G_l$. Namely, the intersection graph is the same for each pair of input graphs. Problem SUNFLOWER SEFE asks whether $\langle G_i(V, E_i) \rangle_{i=1}^{k}$ admit planar drawings $\langle \Gamma_i \rangle_{i=1}^{k}$, respectively, on the same point set such that each edge $e \in G_\cap$ is represented by the same curve in each of $\Gamma_1, \ldots, \Gamma_k$.

Problem SUNFLOWER SEFE has been shown NP-complete for $k = 3$ and $G_\cap$ being a spanning tree [ADN15, ADN14]. Refer also to Section 8.2. However, in our reduction, the produced instances of the problem will always have the property that $G_\cap$ is biconnected and each graph G_i ($1 \leq i \leq k$) is composed only of the edges of $G_\cap$ plus a single edge. In this setting, a polynomial-time algorithm can be easily derived from the known algorithms [ADF+12, BKR13a, HJL13] that solve in polynomial time the case in which $k = 2$ and $G_\cap$ is biconnected.

Algorithm 1-BEND-BICONNECTED-DECISION. The algorithm tests whether S admits an embedding $\mathcal{E}$ such that the end-vertices of each edge $e \in E \setminus W$ belong to the same face of $\mathcal{E}$ in two steps, as follows.

First, instance $\langle G(V, E), S(V, W) \rangle$ is reduced to an instance $\langle G_1, \ldots, G_{|E \setminus W|} \rangle$ of SUNFLOWER SEFE. For each edge e_i with $i = 1, \ldots, |E \setminus W|$, the reduction simply sets $G_i = S \cup e_i$. By construction, the intersection graph $G_\cap$ of the constructed

instance coincides with S (and hence is biconnected) and each graph G_i is composed only of the edges of $G_\cap$ plus a single edge.

Second, the existence of drawings $\Gamma_1, \ldots, \Gamma_{|E \setminus W|}$ of $\langle G_1, \ldots, G_{|E \setminus W|} \rangle$ is tested by means of the algorithm by Bläsius *et al.* [BKR13a], which also constructs such drawings if a SUNFLOWER SEFE exists. The embedding $\mathcal{E}$ of S is obtained by restricting any of the drawings Γ_i to the edges of $G_\cap$ (and hence of S). Note that, in order for $\Gamma_1, \ldots, \Gamma_{|E \setminus W|}$ to be a solution of the SUNFLOWER SEFE instance, the embedding of $G_\cap$ resulting by restricting Γ_i to its edges is the same for every $i = 1, \ldots, |E \setminus W|$.

Theorem 10.7 *Let $G(V, E)$ be a graph and let $S(V, W)$ be a planar biconnected spanning subgraph of G. There exists an $O(|V| + |E \setminus W|)$-time algorithm that decides whether $\langle G, S \rangle$ admits a 1-bend compatible drawing Γ and, in the positive case, computes it on an $O((|V|^2 \cdot |E \setminus W| + |V|) \times |V|)$ grid.*

Proof: First, apply Algorithm 1-BEND-BICONNECTED-DECISION to decide whether $\langle G, S \rangle$ satisfies the condition of Lemma 10.4. If this is the case, apply Algorithm 1-BEND-BICONNECTED-DRAWING to construct a 1-bend compatible drawing Γ of $\langle G, S \rangle$.

CORRECTNESS. The correctness of Algorithm 1-BEND-BICONNECTED-DRAWING has been proved in Lemma 10.4. In order to prove the correctness of Algorithm 1-BEND-BICONNECTED-DECISION, we show that an embedding $\mathcal{E}$ with the required properties exists if and only if the instance of SUNFLOWER SEFE constructed in the first step is positive. Namely, it is easy to observe that in any SUNFLOWER SEFE of the $|E \setminus W|$ constructed graphs, the embedding of S satisfies the condition of Lemma 10.4. For the other direction, it is sufficient to observe that, since each graph G_i contains exactly one edge e_i not belonging to S, once an embedding of S with the property of Lemma 10.4 has been computed, edge e_i can be drawn inside a face of such embedding containing both its end-vertices without intersecting any edge of G_i, thus yielding a SUNFLOWER SEFE of $\langle G_1, \ldots, G_{|E \setminus W|} \rangle$.

TIME AND AREA REQUIREMENTS. The construction of the SUNFLOWER SEFE instance $\langle G_1, \ldots, G_{|E \setminus W|} \rangle$ requires $O(|V| + |E \setminus W|)$ total time, as the description of S does not need to be repeated in each data structure representing a graph G_i. Also, the second step of the algorithm can be performed in $O(|V| + |E \setminus W|)$ total time [ADF+12]. The running time and the area requirements of Algorithm 1-BEND-BICONNECTED-DRAWING have been proved in Lemma 10.4. □

10.5 Conclusions and Open Problems

We introduced a new graph drawing problem, i.e., computing a drawing Γ of a non-planar graph G such that a desired subgraph $S \subseteq G$ is crossing-free in Γ. In the setting where edges are straight-line segments, we showed that Γ does not always exist even if S is a spanning tree of G; also, we provided existential and algorithmic results for meaningful subfamilies of spanning trees and we described a linear-time testing and drawing algorithm when S is a triconnected spanning subgraph. One of the main problems still open in this setting is the following:

Open Problem 1 *Given a graph G and a spanning tree S of G, what is the complexity of deciding whether $\langle G, S\rangle$ admits a straight-line compatible drawing?*

The problem is still open also when S is a biconnected spanning subgraph (for which one can try to extend the characterization of Lemma 10.3) and for subgraphs S that are not necessarily connected and spanning. Hence, the following more general open problem can be stated:

Open Problem 2 *Given a graph G and a subgraph S of G, what is the complexity of deciding whether $\langle G, S\rangle$ admits a straight-line compatible drawing?*

Schaefer proposed a variant of Open Problem 2, where the edges of S are allowed to cross at most h times [Sch14]. In particular, for $h = 1$ he proved that the problem turns out to be as hard as the existential theory of the real numbers.

Another intriguing problem, which is more specific, is to extend the results of Lemmas 10.1 and 10.2. Namely:

Open Problem 3 *Give a characterization of the pairs $\langle G, S\rangle$ that admit a compatible drawing, when G is a complete graph and S is a spanning tree of G.*

Concerning our positive results, our constructive algorithm of a straight-line compatible drawing for the case where S is a spanning caterpillar does not compute integer coordinates for the vertices. Thus, it is natural to ask the following question.

Open Problem 4 *Given a graph G and a spanning caterpillar S of G, does there exist a polynomial-time algorithm that computes a straight-line compatible drawing of $\langle G, S\rangle$ at integer coordinates and with polynomial area?*

In the setting where the edges of G not in S are allowed to bend, we showed that a compatible drawing of $\langle G, S\rangle$ always exists if S is a spanning tree, with different compromises between number of bends per edge and drawing area (see Theorems 10.5 and 10.6). Also, if S is a biconnected spanning subgraph we provided efficient testing and drawing algorithms to compute 1-bend compatible drawings with polynomial area (see Theorem 10.7). However, also in this setting several interesting open problems can be studied; among them we mention the following:

Open Problem 5 *Given a graph G, a non-connected subgraph S of G, and a positive integer k, such that $\langle G, S\rangle$ admits a k-bend compatible drawing, what is the area requirements of constructing such a drawing?*

Observe that, given a pair $\langle G, S\rangle$ such that S is a spanning subgraph of G not necessarily connected, testing whether $\langle G, S\rangle$ admits a k-bend compatible drawing can be done in linear time [DR15].

Open Problem 6 *Is it possible to improve the trade-offs between number of bends per edge and area established by our results when S is a spanning tree of G? For instance, is it possible to reduce the area of 1-bend compatible drawings of $\langle G, S\rangle$ given in Theorem 10.5? Also, what about other popular aesthetic criteria, like for example vertex angular resolution?*

Chapter 11

Planarity of Streamed Graphs

In this chapter[1] we introduce a notion of planarity for graphs that are presented in a streaming fashion. A *streamed graph* is a stream of edges $e_1, e_2, \ldots, e_m$ on a vertex set V. A streamed graph is *ω-stream planar* with respect to a positive integer window size ω if there exists a sequence of planar topological drawings Γ_i of the graphs $G_i = (V, \{e_j \mid i \leq j < i + \omega\})$ such that the common graph $G_\cap^i = G_i \cap G_{i+1}$ is drawn the same in Γ_i and in Γ_{i+1}, for $1 \leq i < m - \omega$. The STREAM PLANARITY problem with window size ω asks whether a given streamed graph is ω-stream planar. We also consider a generalization, where there is an additional *backbone graph* whose edges have to be present during each time step. These problems are related to several well-studied planarity problems.

We show that the STREAM PLANARITY problem is NP-complete even when the window size is a constant and that the variant with a backbone graph is NP-complete for all $\omega \geq 2$. On the positive side, we provide $O(n + \omega m)$-time algorithms for (i) the case $\omega = 1$ and (ii) all values of ω provided the backbone graph consists of one 2-connected component plus isolated vertices and no stream edge connects two isolated vertices. Our results improve on the Hanani-Tutte-style $O((nm)^3)$-time algorithm proposed by Schaefer in [Sch14] for $\omega = 1$.

11.1 Introduction

In this work we consider the following problem concerning the drawing of evolving networks. We are given a stream of edges $e_1, e_2 \ldots, e_m$ with their endpoints in a

[1]The contents of this chapter are a joint work with Ignaz Rutter, appeared in [DR15] and carried out during a visit period in the Department of Applied Mathematics at Charles University of Prague.

vertex set V and an integer *window size* $\omega > 0$. Intuitively, edges of the stream are assigned a fixed "lifetime" of ω time intervals. Namely, for $1 \leq i < |V| - \omega$, edge e_i will *appear* at the i-th time instant and *disappear* at the $(i+\omega)$-th time instant. We aim at finding a sequence of drawings Γ_i of the graphs $G_i = (V, \{e_j \mid i \leq j < i+\omega\})$, for $1 \leq i < |V| - \omega$, showing the vertex set and the subset of the edges of the stream that are "alive" at each time instant i, with the following two properties: (i) each drawing Γ_i is planar and (ii) the drawing of the common graphs $G_\cap^i = G_i \cap G_{i+1}$ is the same in Γ_i and in Γ_{i+1}. We call such a sequence of drawings an *ω-streamed drawing* (ω-SD).

The introduced problem, which we call STREAMED PLANARITY (SP for short), captures the practical need of displaying evolving relationships on the same set of entities. As large changes in consecutive drawings might negatively affect the ability of the user to effectively cope with the evolution of the dataset to maintain his/her mental map, in this model only one edge is allowed to enter the visualization and only one edge is allowed to exit the visualization at each time instant, visible edges are represented by the same curve during their lifetime, and each vertex is represented by the same distinct point. Thus, the amount of relational information displayed at any time stays constant. However, the magnitude of information to be simultaneously presented to the user may significantly depend on the specific application as well as on the nature of the input data. Hence, an interactive visualization system would benefit from the possibility of selecting different time windows. On the other hand, it seems generally reasonable to consider time windows whose size is fixed during the whole animation.

To widen the application scenarios, we consider the possibility of specifying portions of a streamed graph that are alive during the whole animation. These could be, e.g., context-related substructures of the input graph, like the backbone network of the Internet (where edges not in the backbone disappear due to faults or congestion and are later replaced by new ones), or sets of edges directly specified by the user. We call this variant of the problem STREAMED PLANARITY WITH BACKBONE (SPB for short) and the sought sequence of drawings an ω-streamed drawing with backbone (ω-SDB).

Related Work. The problem is similar to on-line planarity testing [DT96b], where one is presented a stream of edge insertions and deletions and has to answer queries whether the current graph is planar. Brandes *et al.* [BDD+12] study the closely related problem of computing planar straight-line grid drawings of trees whose edges have a fixed lifetime under the assumption that the edges are presented one at a time and according to an Eulerian tour of the tree. The main difference, besides using topological rather than straight-line drawings, is that in our model the sequence of edges determining the streamed graph is known in advance and no assumption is made on

the nature of the stream.

It is worth noting that the SP problem can be conveniently interpreted as a variant of the much studied SIMULTANEOUS EMBEDDING WITH FIXED EDGES (SEFE) problem (see [BKR13b] for a recent survey). In short, an instance of SEFE consists of a sequence of graphs $G_1, \ldots, G_k$, sharing some vertices and edges, and the task is to find a sequence of planar drawings Γ_i of G_i such that Γ_i and Γ_j coincide on $G_i \cap G_j$. It is not hard to see that deciding whether a streamed graph is ω-stream planar is equivalent to deciding whether the graphs induced by the edges of the stream that are simultaneously present at each time instant admit a SEFE. Unfortunately, positive results on SEFE mostly concentrate on the variant with $k = 2$, whose complexity is still open, and the problem is NP-hard for $k \geq 3$ [GJP$^+$06]. However, while the SEFE problem allows the edge sets of the input graphs to significantly differ from each other, in our model only small changes in the subsets of the edges of the stream displayed at consecutive time instants are permitted. In this sense, the problems we study can be seen as an attempt to overcome the hardness of SEFE for $k \geq 3$ to enable visualization of graph sequences consisting of several steps, when any two consecutive graphs exhibit a strong similarity.

We note that the ω-stream planarity of the stream $e_1, \ldots, e_m$ on vertex set V and backbone edges S is equivalent to the existence of a drawing of the (multi)graph $G_\cup = (V, \{e_1, \ldots, e_m\} \cup S)$ such that (i) two edges cross only if neither of them is in S and (ii) if e_i and e_j cross, then $|i-j| \geq \omega$. As such the problem is easily seen to be a special case of the WEAK REALIZABILITY problem, which given a graph $G = (V, E)$ and a symmetric relation $R \subseteq E \times E$ asks whether there exists a topological drawing of G such that no pair of edges in R crosses. It follows that SP and SPB are contained in $\mathcal{NP}$ [SSS03]. For $\omega = 1$, the problem amounts to finding a drawing of $G_\cup$, where a subset of the edges, namely the edges of S, are not crossed. This problem has recently been studied under the name PARTIAL PLANARITY [Sch14]; see also Chapter 10 and [ABD$^+$13]. In Chapter 10 and in [ABD$^+$13] the focus is mostly on the geometric version of the STREAMED PLANARITY WITH BACKBONE problem for $\omega = 1$, but it is also noted that the topological variant of this problem can be solved efficiently if the non-crossing edges form a 2-connected graph (see Section 10.4). Recently Schaefer [Sch14] gave an $O((nm)^3)$-time testing algorithm for the general case of PARTIAL PLANARITY exploiting a Hanani-Tutte-style approach. He further suggests to view the relation R of an instance of WEAK REALIZABILITY [Kra98, KLN91] as a conflict graph on the edges of the input graph and to study the complexity subject to structural constraints on this conflict graph.

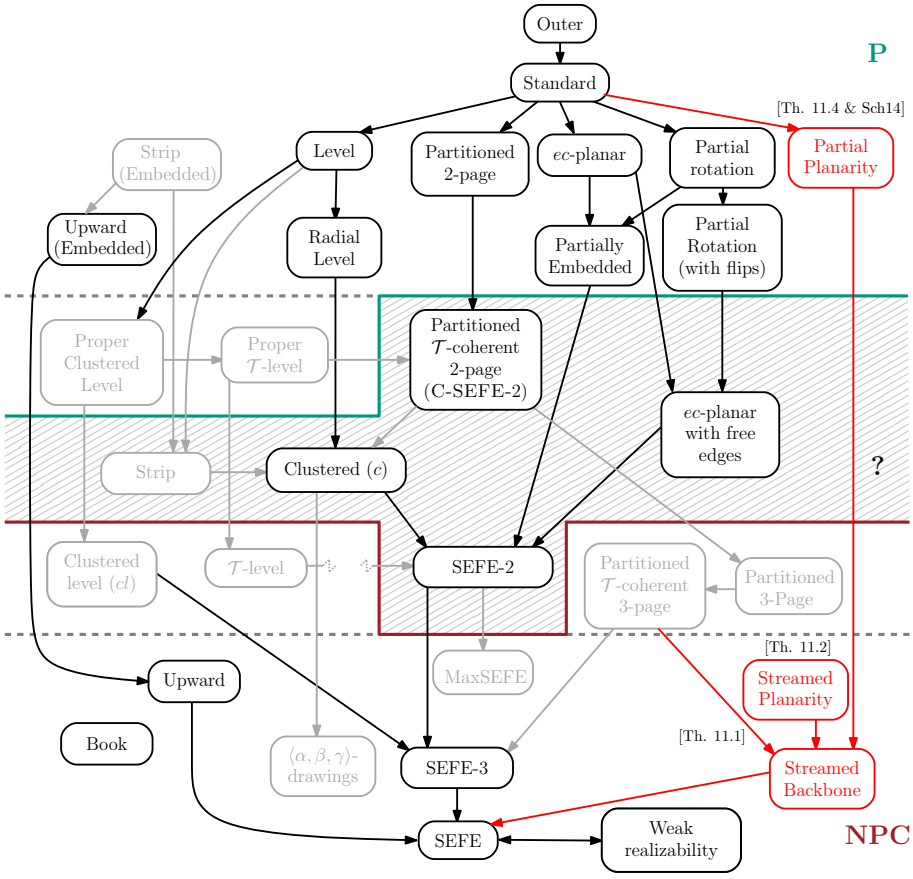


Figure 11.1: View of the schema proposed in the Introduction in which the contributions of this chapter are highlighted by using the red color.

Our Contributions. In this work, we study the complexity of the SP and SPB problems. See Fig. 0.1 in the Introduction and Fig. 11.1 in which the contributions of this chapter are highlighted. In particular, we show the following results.

1. SPB is NP-complete for all $\omega \geq 2$ when the backbone graph is a spanning tree.
2. There is a constant ω_0 such that SP with window size ω_0 is NP-complete.
3. We give an efficient algorithm with running time $O(n+\omega m)$ for SPB when the

backbone graph consists of one 2-connected component plus, possibly, isolated vertices and no stream edge connects two isolated vertices.

4. We give an efficient algorithm for SPB with running time $O(n+m)$ for $\omega = 1$.

It is worth pointing out that the second hardness result shows that WEAK REALIZABILITY is NP-complete even if the conflict graph describing the non-crossing pairs of edges has bounded degree, i.e., every edge may not be crossed only by a constant number of other edges. In particular, this rules out the existence of FPT algorithms with respect to the maximum degree of the conflict graph unless $\mathcal{P} = \mathcal{NP}$.

For the positive results, note that the structural restrictions on the variant for arbitrary values of ω are necessary to overcome the two hardness results and are hence, in a sense, best possible. Moreover, the algorithm for $\omega = 1$ improves the previously best algorithm for PARTIAL PLANARITY by Schaefer [Sch14] (with running time $O((nm)^3)$-time) to linear. Again, since the problem is hard for all $\omega \geq 2$, this result is tight.

11.2 Notation and Preliminaries

In this section we introduce some notations and definitions which will be useful through the chapter.

Given a $(k-1)$-connected graph G with $k \geq 1$, we denote by $k(G)$ the number of its maximal k-connected subgraphs. Recall that, the maximal 2-connected subgraphs are called *blocks*. Also, a k-connected component is *trivial* if it consists of a single vertex.

Contracting an edge (u, v) in a graph G to a new vertex w is the operation of first removing vertices u and v together with their incident edges from G, then adding to G vertex w and making it adjacent to all the edges u and v used to be adjacent to, except for edge (u, v), and finally removing multi-edges.

Let G be a planar graph and let $\mathcal{E}$ be a planar embedding of G. Further, let H be a subgraph of G. We denote by $\mathcal{E}|_H$ the embedding of H determined by $\mathcal{E}$.

For the convenience of the reader, we recall the definition of *simultaneous embedding with fixed edges* (SEFE) for an arbitrary set $\langle G_i(V, E_i)\rangle_{i=1}^k$ of k planar graphs on the same set V of vertices. A SEFE of graphs $\langle G(V, E_i)\rangle_{i=1}^k$ consists of k planar embeddings $\langle \mathcal{E}_i\rangle_{i=1}^k$ such that $\mathcal{E}_i|_{G_{ij}} = \mathcal{E}_j|_{G_{ij}}$, with $G_{ij} = (V, E_i \cap E_j)$ for $i \neq j$. The SEFE problem corresponds to the problem of deciding whether the input graphs $\langle G_i(V, E_i)\rangle_{i=1}^k$ admit a *SEFE*. As described in Section 3.2, if all graphs share the same set of edges (*sunflower intersection*), that is, the graph $G_\cap = (V, E_i \cap E_j)$ is the same for every i and j, with $1 \leq i < j \leq k$, the problem is called SUNFLOWER SEFE and graph $G_\cap$ is the *common graph*.

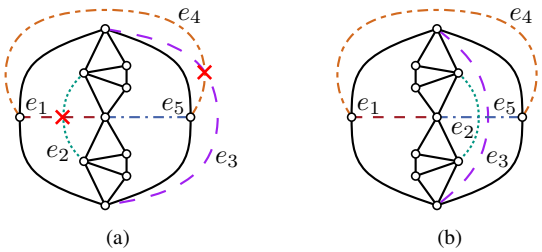


Figure 11.2: Illustration of an instance $\langle G(V,S), E, \Psi \rangle$ of SPB with $\omega = 2$, where G is a 2-connected graph, $E = \{e_i : 1 \leq i \leq 5\}$, and $\Psi(e_i) = i$. Solid edges belong to G. (a) and (b) show different embeddings of G and assignments of the edges in E to the faces of such embeddings. (a) determines a 2-SDB of $\langle G(V,S), E, \Psi \rangle$, while (b) does not.

In the following, we denote a streamed graph by a triple $\langle G(V,S), E, \Psi \rangle$ such that $G(V,S)$ is a planar graph, called *backbone graph*, $E \subseteq V^2 \setminus S$ is the set of edges of a stream $e_1, e_2, \ldots, e_m$, and $\Psi : E \leftrightarrow \{1, \ldots, m\}$ is a bijective function that encodes the ordering of the edges of the stream. Given an instance $I = \langle G(V,S), E, \Psi \rangle$ of SP, we call graph $G_\cup = (V, S \cup E)$ the *union graph* of I. Observe that, if $G_\cup$ has k connected components, then I can be efficiently decomposed into k independent smaller instances, whose Streamed Planarity can be tested independently. Hence, in the following we will only consider streamed graphs with connected union graph. Also, we denote by $\mathcal{Q}$ the set of isolated vertices of G.

Note that, an obvious necessary condition for a streamed graph $\langle G(V,S), E, \Psi \rangle$ to admit an ω-SDB is the existence of a planar combinatorial embedding $\mathcal{E}$ of the backbone graph G such that the endpoints of each edge of the stream lie on the boundary of the same face of $\mathcal{E}$, as otherwise a crossing between an edge of the stream and an edge of G would occur. However, since each edge of the stream must be represented by the same curve at each time, this condition is generally not sufficient, unless $\omega = 1$; see Fig. 11.2.

11.3 Complexity

In the following we study the computational complexity of testing planarity of streamed graphs with and without a backbone graph. First, we show that SPB is NP-complete, even when the backbone graph is a spanning tree and $\omega = 2$. This implies that SUN-

FLOWER SEFE is NP-complete for an arbitrary number of input graphs, even if every graph contains at most $\xi = 2$ exclusive edges. Second, we show that SP is NP-complete even for a constant window size ω. This also has connections to the fundamental WEAK REALIZABILITY problem. Namely, Theorem 11.2 implies the NP-completeness of WEAK REALIZABILITY even for instances $\langle G(V, E), R\rangle$ such that the maximum number of occurrences θ of each edge of E in the pairs of edges in R is bounded by a constant, i.e., for each edge there is only a constant number θ of other edges it may not cross.

These results imply that, unless P=NP, no FPT algorithm with respect to ω, to ξ, or to θ exists for STREAMED PLANARITY (WITH BACKBONE), SEFE, and WEAK REALIZABILITY problems, respectively.

Theorem 11.1 SPB *is NP-complete for $\omega \geq 2$, even when the backbone graph is a tree and the edges of the stream form a matching.*

Proof: The membership in NPfollows from [SSS03]. The NP-hardness is proved by means of a polynomial-time reduction from problem SUNFLOWER SEFE, which has been proved NP-complete for $k = 3$ graphs, even when the common graph is a tree T and the exclusive edges of each graph only connect leaves of the tree [ADN14].

Given an instance $\langle G_i(V, E_i)\rangle_{i=1}^{3}$ of SUNFLOWER SEFE, we construct a streamed graph $\langle G(V, S), E, \Psi\rangle$ that admits an ω-SDB for $\omega = 2$ if and only if $\langle G_i(V, E_i)\rangle_{i=1}^{3}$ is a positive instance of SUNFLOWER SEFE, as follows. To simplify the construction, we first replace instance $\langle G_i(V, E_i)\rangle_{i=1}^{3}$ of SUNFLOWER SEFE with an equivalent instance in which the exclusive edges in $E_1 \cup E_2 \cup E_3$ form a matching, by applying the technique described in [ADF$^+$12]. Then, we perform the reduction starting from such a new instance. Refer to Fig. 11.3.

First, set $G = T$. Then, for $i = 1, 2, 3$ and for each edge $e = (u, v) \in E_i$, add to G a star graph[2] $S(u_e)$ with leaves $u_e^1, \ldots, u_e^q$ and a star graph $S(v_e)$ with leaves $v_e^1, \ldots, v_e^q$ with $q = |E_i| - 1$, and identify the center of $S(u_e)$ with u and the center of $S(v_e)$ with v, respectively. Also, consider the vertex ρ of G corresponding to any internal node of T, add to G vertices s_i, for $i = 1, \ldots, 6$ (*sentinel leaves*), and connect each of such vertices to ρ. Observe that, by construction, G is a tree and $T \subset G$. The sentinel edges will serve as endpoints of edges of the stream, called *sentinel edges*, used to split the stream in three substreams in such a way that no edge of one substream is alive together with an edge of a different substream.

Further, set E can be constructed as follows. For $i = 1, 2, 3$ and for each pair $\langle l, m\rangle$ of edges in E_i, add to E an edge $lm = (u_l^a, v_l^a)$ between a leaf of $S(u_l)$ and

[2] A star graph is a tree with one internal node, called the *central vertex* of the star, and k leaves.

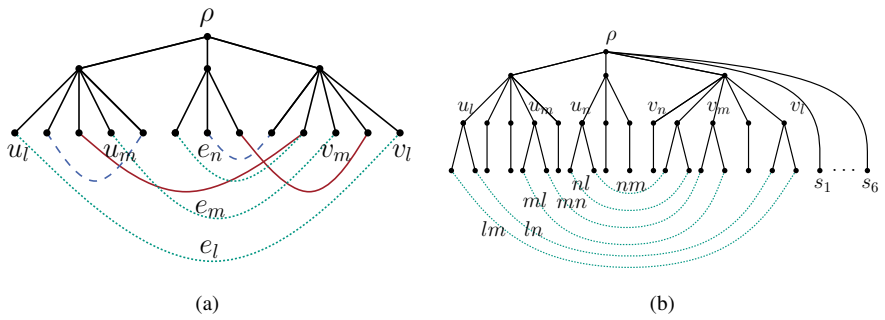


Figure 11.3: Illustration for the proof of Theorem 11.1. (a) Instance $\langle G_i(V, E_i)\rangle_{i=1}^{3}$. (b) Partial representation of instance $\langle G(V, S), E, \Psi\rangle$ containing the edges of G and the edges of the stream constructed starting from pairs of edges of E_3. Edges of T and G are black, edges of G_1, G_2, and G_3 are solid red, dashed blue, and dotted green, respectively.

a leaf of $S(v_l)$ and an edge $ml = (u_m^b, v_m^b)$ between a leaf of $S(u_m)$ and a leaf of $S(v_m)$, respectively, for some $a, b \in 1, 2, \ldots, |E_i| - 1$, in such a way that no two edges in E are incident to the same leaf of G. Observe that, by construction, E is a matching. Also, add to E edges (s_1, s_2), (s_3, s_4), and (s_5, s_6) (*sentinel edges*).

Function Ψ can be defined as follows. First, we construct an auxiliary ordering $\sigma = e_h, \ldots, e_g$ of the edges in E, then we just set $\Psi(e) = \sigma(e)$, for any edge $e \in E$, where $\sigma(e)$ denotes the position of e in σ. To obtain σ, we consider sets E_1, E_2, and E_3 in this order and perform the following two steps. STEP 1: for each pair $\langle l, m\rangle$ of edges in E_i, add to σ edge lm and edge ml. STEP 2: add to σ the sentinel edge $(v_{2(i-1)+1}, u_{2(i-1)+2})$. Observe that, by construction, each common graph $G_\cap^i$ contains the edges of G plus at most two edges lm and ml of the stream with $l, m \in E_i$, for some $i \in \{1, 2, 3\}$.

Observe that, the reduction can be easily performed in polynomial time.

We now shot that $\langle G_i(V, E_i)\rangle_{i=1}^{3}$ admits a SEFE if and only if instance $\langle G(V, S), E, \Psi\rangle$ admits an ω-SDB for $\omega = 2$.

Suppose that $\langle G_i(V, E_i)\rangle_{i=1}^{3}$ admits a SEFE $\langle \mathcal{E}_i\rangle_{i=1}^{3}$. Let $\mathcal{H}$ be the embedding of the common graph T in $\langle \mathcal{E}_i\rangle_{i=1}^{3}$, that is, $\mathcal{H} = \mathcal{E}_1|_T = \mathcal{E}_2|_T = \mathcal{E}_3|_T$. We construct a planar embedding $\mathcal{E}$ of G by defining the rotation scheme of each non-leaf vertex of

G, as follows.

If v is not a leaf of T, then the rotation scheme of v in $\mathcal{E}$ is equal to the rotation scheme of v in $\mathcal{H}$. If $v = u_l$ ($v = v_l$) is the unique neighbor of of any leaf vertex of G, then the rotation scheme of u_l (v_l) can be chosen in such a way that the ordering of the leaves of G that are adjacent to u_l (v_l) is the reverse of the ordering of the leaves of G that are adjacent to v_l (u_l), where the leaves of G that are adjacent to u_l (v_l) and to v_l (u_l) are identified by the corresponding apex. We claim that the constructed embedding $\mathcal{E}$ of G yields an ω-SDB of $\langle G(V,S), E, \Psi\rangle$ for $\omega = 2$. Let $\mathcal{O}$ be the circular ordering of the leaves of T determined by an Eulerian tour of T in $\mathcal{H}$. Also, let $\mathcal{O}'$ be the circular ordering of the leaves of G determined by an Eulerian tour of G in $\mathcal{E}$. Suppose that there exist two edges xy and yx with $|\Psi(xy) - \Psi(yx)| < \omega = 2$ such that the endpoints u_x^i and v_x^i of edge xy and the endpoints u_y^j and v_y^j of edge yx alternate in $\mathcal{O}'$. This implies that the unique neighbors u_x of u_x^i, v_x of v_x^i, u_y of u_y^j, and v_y of v_y^j in T alternate in $\mathcal{O}$. This, in turn, implies a crossing between the two edges x and y of some set E_i. Hence, contradicting the fact that $\langle \mathcal{E}_i \rangle_{i=1}^3$ is a SEFE.

Suppose that $\langle G(V,S), E, \Psi\rangle$ admits an ω-SDB for $\omega = 2$. Let $\mathcal{E}$ be the planar embedding of G in any ω-SDB of $\langle G(V,S), E, \Psi\rangle$. Let $\mathcal{O}$ be the ordering of the leaves of G in an Eulerian tour of G in $\mathcal{E}$. Also, let $\mathcal{O}'$ of the ordering of the leaves of T in an Eulerian tour of T in the embedding $H = \mathcal{E}|_T$. We claim that H yields a SEFE of $\langle G_i(V,E_i)\rangle_{i=1}^3$. Suppose that there exist two edges $x = (u_x, v_x)$ and $y = (u_y, v_y)$ of some set E_i whose endpoints alternate in $\mathcal{O}'$. Consider the two edges $xy = (u_x^p, v_x^p)$ and $yx = (u_y^q, v_y^q)$ of E, with $1 \leq p \leq |E_i^*| - 1$ and $1 \leq q \leq |E_i^*| - 1$. Since the sets of leaves of $S(u_x)$, $S(v_x)$, $S(u_y)$, and $S(v_y)$ appear in $\mathcal{O}$ in the same order as the vertices u_x, v_x, u_y, and v_y appear in $\mathcal{O}'$, the endpoints of xy and yx alternate in $\mathcal{O}'$. Further, by construction, it holds that either $\Psi(xy) = \Psi(yx) + 1$ or $\Psi(yx) = \Psi(xy) + 1$, that is, either edge xy immediately precedes edge yx in the stream or edge yx immediately precedes edge xy in the stream. The above facts then imply a crossing between edge xy and yx of the stream. Hence, contradicting the hypothesis that $\langle G(V,S), E, \Psi\rangle$ admits an ω-SDB for $\omega = 2$.

The above discussion proves the statement for $\omega = 2$. To extend the theorem to any value of $\omega \geq 2$ it suffices to augment $\langle G(V,S), E, \Psi\rangle$ with additional sentinel leaves and sentinel edges. This concludes the proof of the theorem. □

Theorem 11.2 *There is a constant ω_0 such that deciding whether a given streamed graph is ω_0-stream planar is NP-complete.*

Proof: The membership in NPfollows from [SSS03]. In the following we describe a reduction that, given a 3-SAT formula φ, produces a streamed graph that is ω_0-stream planar if and only if φ is satisfiable.

To make things simple, we do not describe the stream, but rather important keyframes. Our construction has the property that edges have a FIFO behavior, i.e., if edge e appears before edge f, then also e disappears before f. This, together with the fact that in each key frame only $O(1)$ edges are visible ensures that the construction can indeed be encoded as a stream with window size $O(1)$. The value ω_0 we use is simply the maximum number of visible edges in any of the key frames. We do not take steps to further minimize ω_0, but even without this, the value produced by the reduction is certainly less than 120, as we estimate at the end of the proof. Sometimes, we wish to wait until a certain set of edges has disappeared. In this case we insert sufficiently many isolated edges into the stream, which does not change the ω_0-planarity of the stream.

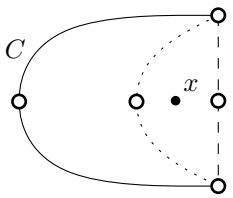
We now sketch the construction. It consists of two main pieces. The first is a cage providing two faces called *cells*, one for vertices representing satisfied literals and one for vertices representing unsatisfied literals. We then present a clause stream for each clause of φ. It contains one literal vertex for each literal occurring in the clause and it ensures that these literal vertices are distributed to the two cells of the cage such that at least one goes in the cell for satisfied literals. Throughout we ensure that none of the previously distributed vertices leaves the respective cell.

Second, we present a sequence of edges that is ω_0-stream planar if and only if the previously chosen distribution of the literal vertices forms a truth assignment. This is the case if and only if any two vertices representing the same literal are in the same cell and any two vertices representing complementary literals of one variable are in distinct cells.

It is clear that, if the constructions work as described, then the resulting streamed graph is ω_0-stream planar if and only if φ is satisfiable. The first part of the stream ensure that from each clause one of the literals must be assigned to the cell containing satisfied literals (i.e. the literal receives the value true). The second part ensures that these choices are consistent over all literals, i.e., these choices actually correspond to a truth assignment of the variables.

Our first step will be the construction of the cage containing the two cells. Since the cage needs to persist throughout the whole sequence, it must be constructed in such a way that it can be "kept alive" over time by presenting new edges. Note that it does not suffice to repeatedly present edges that are parallel to existing ones, as they may be embedded differently, and hence over time allow isolated vertices to move through obstacles; see Fig. 11.4. We first present a construction that behaves like an edge that can be "renewed" without changing its drawing too much. We call it *persistent edge*.

Let u and v be two vertices. A persistent edge between u and v consists of the four vertices a, b, c, d, each lying on a path of length 2 from u to v. Additionally, a is connected to b and b is connected to c. Initially, we also have insert the edge b, c



(a)

Figure 11.4: Cycle C (solid and dashed edges) contains vertex x in its interior. The dashed edges leave the sliding window soon. Presenting a new path (dotted) parallel to the old path does not ensure that x ends up in the interior of the resulting cycle C' (solid and dotted edges).

to enforce a unique planar embedding. However, once it leaves the sliding window it does not get replaced. Figure 11.5(a) shows a persistent edge where the thickness of the edge visualizes the time until an edge leaves the sliding window. The thicker the edge the longer it stays. Once the edge bc has been removed, but before any of the other edges disappear, we present in the stream the edges ub', vb' and bb' as well as uc', vc' and cc', where b' and c' are new vertices; see Fig. 11.5(b). Note that there is a unique way to embed them into the given drawing. After the edges ua, av leave the sliding window, b takes over the role of a and b' takes over the role of b. Similarly, after the edges ud and dv leave the sliding window, c takes over the role of d and c' takes over the role of c; see Fig. 11.5(c). By presenting six new edges in regular intervals, the persistent edge essentially keeps its structure. In particular, we know at any point in time which vertices are incident to the inner and outer face. For simplicity we will not describe in detail when to perform this book keeping. Rather, we just assume that the sliding window is sufficiently large to allow regular book keeping. For example, before each of the steps described later, we might first update all persistent edges, then present the gadget performing one of the steps, then update the persistent edges gain, and finally wait for the gadget edges to be removed from the sliding window again.

Next, we describe the cage. Conceptually, it consists of two cycles of length 4, on vertices a, b, c, v^+ and a, b, c, v^-, respectively. However, the edges are actually persistent edges; see Fig. 11.6(a). The interior faces f^+ and f^- of the two cycles are the positive and negative literal faces, respectively. Note that at any point in time only a constant number of edges are necessary for the cage.

Before we describe the clause gadget, which is the most involved part of the construction, we briefly show how to perform the test for the end of sequence. Namely,

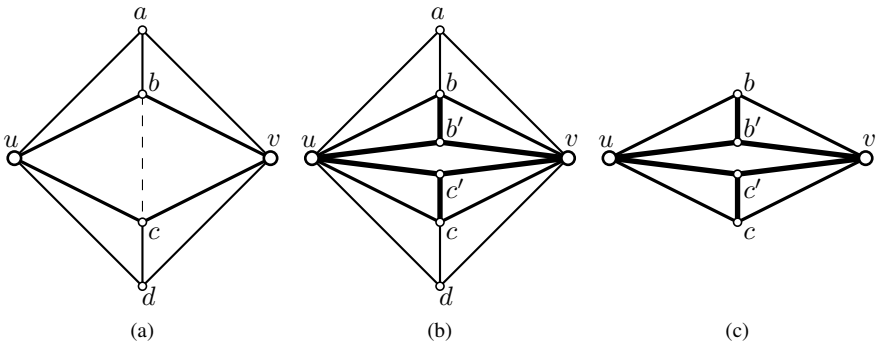


Figure 11.5: A persistent edge. The thickness of the edges indicates how long the edge stays in the sliding window. The thinner the edge the earlier it leaves the window. (a) The initial configuration; the dashed edge bc dissolves first. It is used only once to initially enforce a unique planar embedding. (b) New vertices b' and c' with neighbors u, b, v and u, c, v, respectively, are introduced. Starting from the embedding in (a) the embedding is uniquely defined. (c) After the edges incident to a and d disappear, the drawing has again the same structure as in (a). Repeating this cycle hence preserves the edge. Since edges are embedded only in the interior of the gadget vertices that are embedded outside the persistent edge cannot traverse it.

assume that we have a set $V' \subseteq V$ of literal vertices, and each of them is contained in one of the two literal faces. More formally, for each clause $c_i \in \varphi$ and for each Boolean variable x, set V' contains a literal vertex x_i, if $x \in c_i$, or a literal vertex $\overline{x_i}$, if $\overline{x} \in c_i$. To check whether two literal vertices x_i and x_j corresponding to a variable x are in the same face, it suffices to present an edge between them in the stream, then wait until that edge leaves the sliding window, and continue with the next pair; see Fig 11.6(b). Of course, in the meantime we may have to refresh the persistent edges. Similarly, if we wish to check that literal vertices $\overline{x_i}$ and x_j are in distinct faces, we make use of the fact that the two cycles forming the cage share two edges, and hence three vertices a, b and c. We present in the stream the complete bipartite graph on the vertices $\{\overline{x_i}, x_j\}$ and $\{a, b, c\}$. Clearly, this can be drawn in a planar way if and only if $\overline{x_i}$ and x_j are in distinct faces; see Fig. 11.6(c). Again, it may be necessary to wait until these edges leave the sliding window before the next test can be performed.

Finally, we describe our clause gadget; see Fig. 11.7 for an illustration. First, we present the clause gadget as it is shown in Fig. 11.7(a). The literal vertices are large and solid, their corresponding indicator vertices are represented by large empty

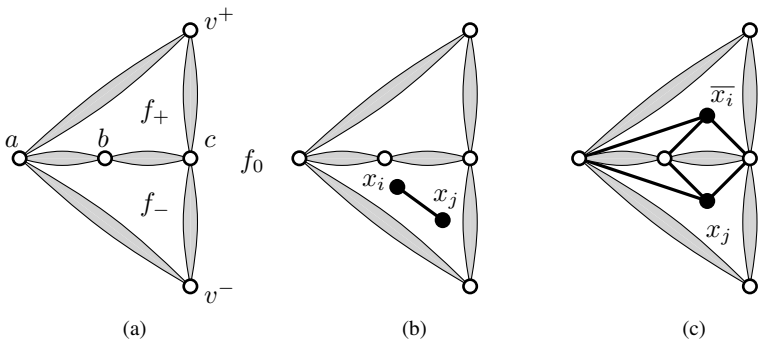


Figure 11.6: (a) The cage, the thick gray edges are persistent edges and are refreshed at regular intervals. After presenting all clause sequences, the faces f^+ and f^- will contain the literal vertices corresponding to satisfied and unsatisfied literal vertices, respectively. (b) Edges used to check whether two literal vertices x_i and x_j are in the same face. (c) Edges used to check whether literal vertices $\overline{x_i}$ and x_j are in distinct faces.

disks. The edges are ordered in the stream such that the three edges connecting a literal vertex to its indicator are presented first, i.e., they also leave the sliding window first. The remaining three edges incident to the literals are drawn last so that they remain present longest. Observe that the embedding of the clause without the literal and indicator vertices is unique; we call this part of the clause the *frame*. Each literal vertex may choose among two possible faces of the frame where it can be embedded. Either close to the center or close to the boundary. The faces in the center are shaded light gray, the faces on the boundary are shaded or tiled in a darker gray in Fig. 11.7(a).

We now first wait until the edges between literal vertices and their indicators leave the sliding window. Now the following things happen. First, the thin dotted and dashed edges leave the sliding window. Immediately afterwards, we present in the stream paths of length 2 that replace these edges, so the frame essentially remains as it is shown. However, after this step, the indicator vertex of any literal that was embedded in the face close to the center may be in any of the faces shaded in light gray in Fig. 11.7(b). Now, first the thick dotted edges leave the sliding window and are immediately replaced by parallel paths. Afterwards, the thick dashed edges leave the sliding window and are immediately replaced by parallel paths. Again, the frame remains essentially present. This allows the indicator vertices of literals that were embedded on the outer face to traverse into the faces indicated in Fig. 11.7(c). Note

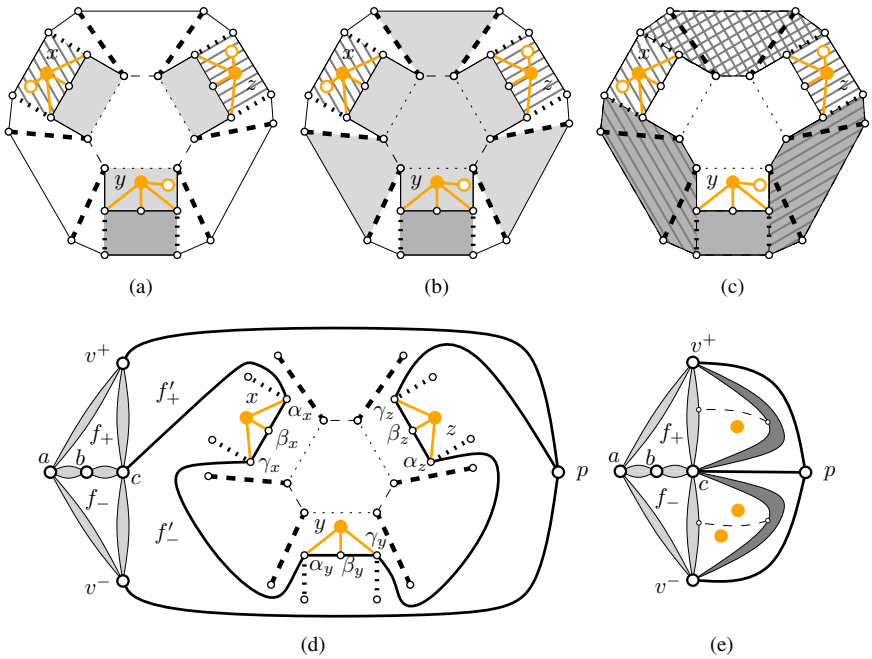


Figure 11.7: Illustration of the clause sequence. (a) Initial embedding of the clause. (b), (c) faces indicator vertices can reach if they are embedded in the face close to the center and close to the boundary, respectively. (d), separating the vertices corresponding to satisfied and unsatisfied literals into two distinct faces. (e) Integrating the now separated literal vertices into the corresponding faces of the cage.

that, if all literal vertices were embedded in the face close the boundary, then there is no face of the frame that can simultaneously contain them at this point. If however, at least one of them was embedded in the face close to the center, then there is at least one face of the frame that can contain all the vertices simultaneously. We now include in the stream a triangle on the three indicator vertices. This triangle can be drawn without crossing edges of the frame if and only if the three vertices can meet in one face, which is the case if and only if at least one indicator vertex, and hence also its corresponding literal vertex, was embedded close to the center. Now we wait until the edges of the clause, except for those incident to the literal vertices and the paths that were renewed have vanished; see Fig. 11.7(d).

Let now p be a new vertex, and denote the neighbors of the literal vertex x by α_x, β_x and γ_x, and similarly for y and z. We now connect v to the cage by present the edges v^-v and v^+v as well as edges forming a path from c to p that, starting from p, first visits $\alpha_x, \beta_x, \gamma_x$, then $\alpha_y, \beta_y, \gamma_y$, and finally $\alpha_z, \beta_z, \gamma_z$. Observe that the fact that p has disjoint paths to $v^-, v+$ and v containing the α_h, β_h and γ_h, with $h \in \{x, y, z\}$, ensures that, what remains of the clause gadget must be (and hence must have been all the time) embedded in the outer face of the cage. We assume without loss of generality that the path containing the α_h, β_h and γ_h, with $h \in \{x, y, z\}$, is not incident to the outer face. Again, we consider the edges incident to the literal vertices not as part of the construction. Then the path is incident to precisely two faces, which are adjacent to the literal faces of the cage. Denote the one incident to f_+ by f'_+ and the one incident to f_- by f'_-; see Fig. 11.7(d). Due to the traversal, we have that a literal vertex v is contained in f'_+ if and only if it was embedded in the face close to the center in the clause, which means that the corresponding literal was satisfied. Otherwise, it is embedded in f'_-. It now remains to enclose the literal vertices into the corresponding face of the cage without letting escape any of the literal vertices already embedded there.

First, we wait until all edges incident to the literal vertices have left the sliding window, i.e., they become isolated. Then, we present two new persistent edges parallel to the existing persistent edges v^+c and v^-c, respectively; see Fig. 11.7(e), where the new persistent edges are shaded dark gray. To ensure that the embedded is indeed as shown in Fig. 11.7(e), we one boundary vertex of each new persistent edge to a vertex on the outer boundary of the persistent edge it is parallel to (dashed lines in Fig. 11.7(e)). The new parallel edges replace the old persistent edges of the cage, and we wait until they have dissolved. Clearly, no vertex from an internal face of the cage can escape as the new persistent edges are embedded in the outer face of the cage. To ensure that the literal vertices must indeed be embedded in the literal faces, we present the edges bx, by and bz. Finally, we wait until these edges vanish again. Then we are ready for the next clause sequence or for the final checking sequence.

The above description produces for a given 3SAT formula φ produces, for a sufficiently large (but constant!) ω_0 a stream S_φ one some vertex set V_φ such that φ is satisfiable if and only if S_φ is ω_0-stream planar. In the first part of the stream, in any sequence of corresponding planar embedding, the literals of each clause, represented by vertices, are transferred to two interior faces of the cage such that for each clause at least one literal vertex is transferred to the face representing satisfied literals. This models the fact that each clause must contain at least one satisfied literal. In the second part, a sequence of edges is presented that is ω_0-planar if and only if the previously produced distribution of literals to the positive and negative faces of the cage corresponds to a truth assignment of the underlying variables. The construction can

clearly be performed in polynomial time.

We now briefly estimate the window size ω_0. The largest number of edges that are simultaneously important in our construction occurs when presenting a clause gadget. A clause gadget has 48 edges, and it is simultaneously visible with four persistent edges, each of which may use up to 16 edges immediately after they have refreshed. Hence a window size of $\omega_0 = 112$ suffices for the construction. □

11.4 Algorithms for ω-Stream Drawings with Backbone

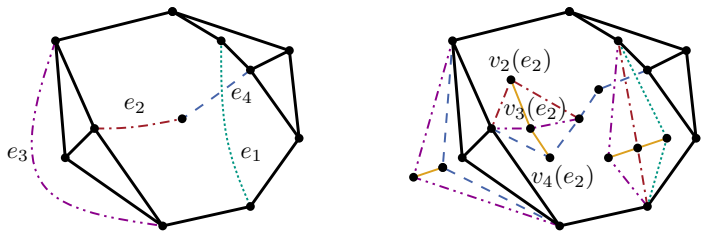
In this section, we describe a polynomial-time decision algorithm for the case that the backbone graph consists of a 2-connected component plus, possibly, isolated vertices with no edge of the stream connecting two isolated vertices. We call instances satisfying these properties *star instances*, as the isolated vertices are the centers of edge disjoint star subgraphs of the union graph (see Section 11.4). Observe that, the requirement of the absence of edges of the stream between the isolated vertices of a star instance seems to be quite a natural restriction. In fact, as proved in Theorem 11.2, dropping this restriction makes the STREAMED PLANARITY problem computationally tough. This algorithm will also serve as a subprocedure to solve the SPB problem for $\omega = 1$ with no restrictions on the backbone graph (see Section 11.4).

Star Instances

In this section we describe an efficient algorithm to test the existence of an ω-SDB for star instances (see Fig. 11.8(a)). The problem is equivalent to finding an embedding $\mathcal{E}$ of the unique non-trivial 2-connected component β of G and an assignment of the edges of the stream and of the isolated vertices of G to the faces of $\mathcal{E}$ that yield a ω-SDB.

Lemma 11.1 *Let $\langle G(V,S), E, \Psi\rangle$ be a star instance of* SPB *and let ω be a positive integer window size. There exists an equivalent instance $\langle G_i(V,E_i)\rangle_{i=1}^{m+1}$ of* SUNFLOWER SEFE *such that the common graph $G_\cap$ consists of disjoint 2-connected components. Further, instance $\langle G_i(V,E_i)\rangle_{i=1}^{m+1}$ can be constructed in $O(n+\omega m)$ time.*

Proof: Given a star instance $\langle G(V,S), E, \Psi\rangle$ of SPB we construct an instance $\langle G_i(V,E_i)\rangle_{i=1}^{m+1}$ of SUNFLOWER SEFE that admits a SEFE if and only if instance $\langle G(V,S), E, \Psi\rangle$ admits an ω-SDB, as follows. Refer to Figs 11.8(a) and 11.8(b) for an example of the construction.



(a) (b)

Figure 11.8: (a) A star instance with stream edges $E = \{e_i : 1 \leq i \leq 4\}$, $\Psi(e_i) = i$, and $\omega = 3$. (b) A SEFE of the instance of SUNFLOWER SEFE obtained as described in Lemma 11.1 where $G_\cup$ is drawn with thick solid black edges, exclusive edges of G_i are drawn with the same style as edge e_i and exclusive edges of $G_{m+1} = G_5$ are drawn as yellow solid curves. Vertices in $D(e_2) = \{v_2(e_2), v_3(e_2), v_4(e_2)\}$ are also shown.

Initialize graph $G_\cap$ to the backbone graph G. Also, for every edge $e \in E$, add to $G_\cap$ a set of vertices $D(e) = \{v_i(e) \mid \Psi(e) \leq i < \min(\Psi(e) + \omega, m + 1)\}$. Observe that, since $\langle G(V, S), E, \Psi\rangle$ is a star instance, graph $G_\cap$ contains a single non-trivial 2-connected component β, plus a set of trivial 2-connected components consisting of the isolated vertices in $\mathcal{Q} \cup \bigcup_{e\in E} D(e)$.

For $i = 1, \ldots, m$, graph G_i contains all the edges and the vertices of $G_\cap$ plus a set of edges defines as follows. For each edge $e = (u, v) \in E$ such that $0 \leq i - \Psi(e) < \omega$, add to $E(G_i)$ edges $(u, v_i(e))$ and $(v_i(e), v)$. From a high-level view, graphs G_i, with $i = 1, \ldots, m$, are defined in such a way to enforce the same constraints on the possible embeddings of the common graph as the constraints enforced by the edges of the stream on the possible embeddings of the backbone graph.

Finally, graph G_{m+1} contains all the edges and the vertices of $G_\cap$ plus a set of edges defined as follows. For each edge $e \in E$, add to E_{m+1} edges $(v_{\Psi(e)}(e), v_k(e))$, with $\Psi(e) < k < min(\Psi(e) + \omega, m + 1)$. Observe that, in any planar drawing Γ_{m+1} of G_{m+1}, vertices $v_k(e)$ lie inside the same face of Γ_{m+1}, for any edge $e \in E$. The aim of graph G_{m+1} is to combine the constrains imposed on the embedding of the backbone graph by each graph G_i, with $i = 1, \ldots, m$, in such a way that, for each edge $e \in E$, the edges of set $D(e)$ are embedded in the same face of the backbone graph.

Hereinafter, given a positive instance $\langle G_i(V, E_i)\rangle_{i=1}^{m+1}$ of SEFE with the above properties, we denote the corresponding SEFE $\langle \Gamma_i\rangle_{i=1}^{m+1}$ by $\langle \mathcal{E}_i, A_i\rangle_{i=1}^{m+1}$, where $\mathcal{E}_i$

represents the embedding of β in Γ_i and $A_{\mathcal{E}_i}$ represents the assignment of the isolated vertices and of the exclusive edges of graph G_i in Γ_i to the faces of $\mathcal{E}_i$, for $i = 1, \ldots, m+1$. Similarly, given a positive star instance $\langle G(V,S), E, \Psi \rangle$ of SPB we denote the corresponding ω-SDB Γ by $\langle \mathcal{E}, A_{\mathcal{E}} \rangle$, where $\mathcal{E}$ represents the embedding of the unique non-trivial 2-connected component β of G in Γ and $A_{\mathcal{E}}$ represents the assignment of the isolated vertices of G and of the edges of the stream to the faces of $\mathcal{E}$ in Γ. More formally, $A_{\mathcal{E}} \colon E \cup \mathcal{Q} \to F(\mathcal{E})$, where $F(\mathcal{E})$ denotes the set of facial cycles of $\mathcal{E}$.

Suppose that $\langle G_i(V,E_i) \rangle_{i=1}^{m+1}$ is a positive instance of SEFE, that is, $\langle G_i(V,E_i) \rangle_{i=1}^{m+1}$ admits a SEFE $\langle \mathcal{E}_i, A_i \rangle_{i=1}^{m+1}$. We show how to construct a solution $\langle \mathcal{E}, A_{\mathcal{E}} \rangle$ of instance $\langle G(V,S), E, \Psi \rangle$.

Since $\langle \mathcal{E}_i, A_i \rangle_{i=1}^{m+1}$ is a SEFE and $\beta \in G_\cap$, we have that $\mathcal{E}_i = \mathcal{E}_j$, with $1 \leq i < j \leq m+1$. We set the embedding $\mathcal{E}$ of β to $\mathcal{E}_1$.

Further, for every edge $e \in E$, we set $A_{\mathcal{E}}(e)$ to the face of $\mathcal{E}_1$ vertex $v_{\Psi(e)}(e)$ is placed inside in Γ_1, that is, $A_{\mathcal{E}}(e) = A_{\mathcal{E}_1}(v_{\Psi(e)}(e))$. Similarly, for every isolated vertex $v \in \mathcal{Q}$, we set $A_{\mathcal{E}}(v)$ to the face of $\mathcal{E}_1$ vertex v is placed inside in Γ_1, that is, $A_{\mathcal{E}}(v) = A_{\mathcal{E}_1}(v)$.

We need to prove that $\mathcal{E}$ is a planar embedding of β and that no crossing occurs neither between an edge in E and an edge in β nor between two edges $e_i \in E$ and $e_j \in E$, with $i < j$ and $\Psi(e_j) - \Psi(e_i) < \omega$. Observe that, since $\langle \mathcal{E}_i, A_i \rangle_{i=1}^{m+1}$ is a SEFE, the embedding $\mathcal{E}_i$ of β in Γ_i is planar. As $\mathcal{E}$ coincides with $\mathcal{E}_1$, it follows that $\mathcal{E}$ is also planar. Assume that there exists a crossing between an edge $e \in E$ and an edge of β. This implies that there exists in $\Gamma_{\Psi(e)}$ a path $p^* = (u, v_{\Psi(e)}(e), v)$ connecting two vertices of u and v of β that are incident to different faces of $\mathcal{E}_{\Psi(e)}$. Further, assume that there exists a crossing between an edge $e_i \in E$ and an edge $e_j \in E$ with $\Psi(e_i) < \Psi(e_j)$ such that $\Psi(e_j) - \Psi(e_i) < \omega$ inside the same face f of $\mathcal{E}$. This implies that there exists in $G_{\Psi(e_i)}$ a crossing between a path $p' = (a, \ldots, v_{\Psi(e_i)}(e_i), \ldots, b)$ and $p'' = (c, \ldots, v_{\Psi(e_i)}(e_j), \ldots, d)$ only containing exclusive edges of $G_{\Psi(e_i)}$ such that a, c, b, and d appear in this order in the face of $\mathcal{E}_{\Psi(e_i)}$ corresponding to f. Thus, both assumptions contradict the fact that $\langle G(V,S), E, \Psi \rangle$ admits an ω-SDB.

Suppose that $\langle G(V,S), E, \Psi \rangle$ admits an ω-SDB, that is, there exist a planar embedding $\mathcal{E}$ of β and an assignment function $A_{\mathcal{E}} : E \cup \mathcal{Q} \to F(\mathcal{E})$ such that, for any two paths $p' = (a, \ldots, b)$ and $p'' = (c, \ldots, d)$ with $\{a, b, c, d\} \in \beta$ and $\Psi(e_j) - \Psi(e_i) < \omega$, for every edge $e_i \in p'$ and $e_j \in p''$ with $i < j$, it holds that $A_{\mathcal{E}}(e_i) \neq A_{\mathcal{E}}(e_j)$. We show how to construct a SEFE $\langle \mathcal{E}_i, A_i \rangle_{i=1}^{m+1}$ of $\langle G_i(V,E_i) \rangle_{i=1}^{m+1}$.

For $i = 1, \ldots, m+1$, we set the embedding $\mathcal{E}_i$ of β to $\mathcal{E}$. For $i = 1, \ldots, m+1$ and for each edge $e \in E$, we assign each vertex $v_k(e) \in D(e)$ to the face of $\mathcal{E}_i$ that corresponds to the face of $\mathcal{E}$ edge e is assigned to, that is, $A_{\mathcal{E}_i}(v_k(e)) = A_{\mathcal{E}}(e)$.

Also, for each edge $e = (u, v) \in E$, we assign edges $(u, v_k(e))$ and $(v_k(e), v)$ to face $A_{\mathcal{E}_k}(v_k(e))$, with $\Psi(e) \leq k < min(\Psi(e) + \omega, m + 1)$. Further, for each edge $e = (u, v) \in E$, we assign edges $(v_{\Psi(e)}, v_k(e))$ to face $A_{\mathcal{E}_{m+1}}(v_k(e))$, with $\Psi(e) < k < min(\Psi(e) + \omega, m + 1)$. Finally, for $i = 1, \ldots, m + 1$ and for each vertex $v \in \mathcal{Q}$, we set $A_{\mathcal{E}_i}(v) = A_{\mathcal{E}}(v)$.

In order to prove that $\langle \mathcal{E}_i, A_i \rangle_{i=1}^{m+1}$ is a SEFE of $\langle G_i(V, E_i) \rangle_{i=1}^{m+1}$ we show that (i) $\mathcal{E}_i$ is a planar embedding of $\beta \in G_i$ (ii) all embeddings $\mathcal{E}_i$ coincide, (ii) there exists no crossing in Γ_i involving the exclusive edges of any graph G_i, and (iv) each isolated vertex v of $G_\cap$ is such that $A_{\mathcal{E}_i}(v) = A_{\mathcal{E}_j}(v)$, with $i \neq j$. Since $\mathcal{E}$ is planar by hypothesis and since $\mathcal{E}_i = \mathcal{E}$, condition (i) is trivially verified. Further, by construction, conditions (ii) and (iv), are also satisfied. Assume that condition (iii) does not hold. In this case, either an exclusive edge $(v_i(e), w)$ of G_i crosses an edge of β or there exists a crossing between two exclusive edges $(v_i(e_1), p)$ and $(v_i(e_2), q)$ of G_i inside the same face of $\mathcal{E}_i$. In the former case, there must exists in G_i a path $p_0 = (a, v_i(e), b)$ composed of exclusive edges of G_i connecting two vertices $a, b \in \beta$ (not necessarily different from w) that lie on the boundary of different faces of $\mathcal{E}_i$. However, this would imply that $G_\cup$ contains a path $p_0^* = (a, \ldots, b)$ containing edge e and only consisting of edges e_k with $0 \leq i - \Psi(e_k) < \omega$, whose endpoints a and b lie on different faces of $\mathcal{E}$. In the latter case, there must exist two vertex-disjoint paths $p_1 = (a, \ldots, v_i(e_1), \ldots, b)$ and $p_2 = (c, \ldots, v_i(e_2), \ldots, d)$ of exclusive edges of G_i contained in a face f of $\mathcal{E}_i$ connecting vertices $a, b \in f$ and $c, d \in f$, respectively, such that a, c, b, and d appear in this order along f. However, this would imply that $G_\cup$ contains two paths $p_1^* = (a, \ldots, b)$ and $p_2^* = (c, \ldots, d)$ with endpoints in β containing edges e_1 and e_2, respectively, and only containing edges e_k in E with $0 \leq i - \Psi(e_k) < \omega$ that lie inside the face f^* of $\mathcal{E}$ corresponding to face f of $\mathcal{E}_i$ and whose endpoints alternate along the boundary of f^*. Thus, both assumptions contradict the fact that $\langle G(V, S), E, \Psi \rangle$ admits an ω-SDB.

It is easy to see that instance $\langle G_i(V, E_i) \rangle_{i=1}^{m+1}$ can be constructed in time $O(n + \omega m)$. In fact, the construction of the common graph $G_\cap$ takes $O(n)$-time, since the backbone graph G is planar. Also, each graph G_i can be encoded as the union of a pointer to the encoding of $G_\cap$ and of the encoding of its exclusive edges. Further, each graph G_i, with $i = 1, \ldots, m$, has at most ω exclusive edges, and graph G_{m+1} has at most ωm exclusive edges. This concludes the proof of the lemma. □

Lemma 11.1 provides a straight-forward technique to decide whether a star instance $\langle G(V, S), E, \Psi \rangle$ of SPB admits a ω-SDB. First, transform instance $\langle G(V, S), E, \Psi \rangle$ into an equivalent instance $\langle G_i(V, E_i) \rangle_{i=1}^{m+1}$ of SEFE of $m+1$ graphs with sunflower intersection and such that the common graph consists of disjoint 2-connected components, by applying the reduction described in the proof of Lemma 11.1.

Then, apply to instance $\langle G_i(V, E_i)\rangle_{i=1}^{m+1}$ the algorithm by Bläsius *et al.* [BKR13a] that tests instances of SEFE with the above properties in linear time. Thus, we obtain the following theorem.

Theorem 11.3 *Let $\langle G(V, S), E, \Psi\rangle$ be an star instance of* SPB. *There exists an $O(n+\omega m)$-time algorithm to decide whether $\langle G(V, S), E, \Psi\rangle$ admits an ω-SDB.*

Unit window size

In this section we describe a polynomial-time algorithm to test whether an instance $\langle G(V, S), E, \Psi\rangle$ of SPB admits an ω-SDB for $\omega = 1$. Observe that, in the case in which $\omega = 1$, the SPB problem equals to the problem of deciding whether an embedding of the backbone graph exists such that the endpoints of each edge of the stream lie on the boundary of the same face of such an embedding.

Let $\mathcal{G}_1, \ldots, \mathcal{G}_{1(G)}$ be the connected components of the backbone graph G. Given an embedding $\mathcal{E}$ of G, we define the set $F(\mathcal{E})$ of facial cycles of $\mathcal{E}$ as the union of the facial cycles of the embeddings $\mathcal{E}_i = \mathcal{E}|_{\mathcal{G}_i}$ of each connected component $\mathcal{G}_i$ of G in $\mathcal{E}$. We first prove an auxiliary lemma which allows us to focus our attention only on instances whose backbone graph contains at most one non-trivial connected component.

Lemma 11.2 *Let $\langle G(V, S), E, \Psi\rangle$ be an instance of* SPB. *There exists a set of instances $\langle G(V_i, S_i), E_i, \Psi_i\rangle$ whose backbone graph $G(V_i, S_i)$ contains at most one non-trivial connected component $\mathcal{G}_i$ such that $\langle G(V, S), E, \Psi\rangle$ admits a ω-SDB with $\omega = 1$ if and only if all instances $\langle G(V_i, S_i), E_i, \Psi_i\rangle$ admit a ω-SDB with $\omega = 1$. Further, such instances can be constructed in $O(n + m)$ time.*

Proof: We construct instances $\langle G(V_i, S_i), E_i, \Psi_i\rangle$ starting from $G_\cup$ in two steps. To ease the description, we assume that each vertex $v \in V$ is initially associated with an index $l(v)$ corresponding to the connected component of G vertex v belongs to, that is, $l(v) = i$ if $v \in V(\mathcal{G}_i)$. First, we recursively contract each edge (u, v) of $G_\cup$ with $\{u, v\} \subseteq V(\mathcal{G}_i)$ to a single vertex w and set $l(w) = i$, for $i = 1, \ldots, 1(G)$. Thus, obtaining an auxiliary graph H on $1(G)$ vertices. Then, we obtain instances $\langle G(V_i, S_i), E_i, \Psi_i\rangle$ from H by recursively uncontracting each vertex w with $l(w) = i$, for $i = 1, \ldots, 1(G)$. Note that, by construction, $\mathcal{G}_i \subseteq G(V_i, S_i)$.

Observe that, the construction of H requires $O(n + m)$ time. Further, the construction of each instance $\langle G(V_i, S_i), E_i, \Psi_i\rangle$ can be performed in $O(n_i + m_i)$ time, where $n_i = |V(\mathcal{G}_i)|$ and m_i is the number of edges in E that are incident to a vertex of G_i, which sums up to $O(n + m)$ time in total for all $1 \leq i \leq 1(G)$. Thus, proving the $O(n + m)$ running time of the construction.

The necessity is trivial. In order to prove the sufficiency, assume that all instances $\langle G(V_i, S_i), E_i, \Psi_i \rangle$ admit a ω-SDB for $\omega = 1$. Intuitively, a 1-SDB Γ of the original instance can be obtained, starting from a 1-SDB Γ_i of any $\langle G(V_i, S_i), E_i, \Psi_i \rangle$, by recursively replacing the drawing of each isolated vertex $v_j \in \mathcal{Q}_i$ with the 1-SDB Γ_j of $\langle G(V_j, S_j), E_j, \Psi_j \rangle$ (after, possibly, promoting a different face to be the outer face of Γ_j) . For a complete example, see Fig. 11.9.

The fact that Γ is a 1-SDB of $\langle G(V, S), E, \Psi \rangle$ derives from the fact that each Γ_i is a 1-SDB of $\langle G(V_i, S_i), E_i, \Psi_i \rangle$, that in a 1-SDB crossings among edges in E do not matter, and that, by the connectivity of the union graph, the assignment of the isolated vertices in $\mathcal{Q}_i$ to the faces of the embedding $\mathcal{E}_i$ of $\mathcal{G}_i$ in Γ_i must be such that any two isolated vertices connected by a path of edges of the stream E_i lie inside the same face of $\mathcal{E}_i$. In the following, we prove this direction more formally.

We denote by $(\mathcal{E}_i, C_{\mathcal{E}_i})$ the solution of instance $\langle G(V_i, S_i), E_i, \Psi_i \rangle$, where $\mathcal{E}_i$ is a planar embedding of $\mathcal{G}_i$ and $C_{\mathcal{E}_i} : F(\mathcal{E}_i) \rightarrow 2^{\mathcal{Q}_i}$ is an assignment of the set of isolated vertices $\mathcal{Q}_i$ of $G(V_i, S_i)$ to the set of faces of $\mathcal{E}_i$, denoted by $F(\mathcal{E}_i)$. We now show how to extend the solutions $(\mathcal{E}_i, C_{\mathcal{E}_i})$ of instances $\langle G(V_i, S_i), E_i, \Psi_i \rangle$, with $i = 1, \ldots, 1(G)$, to a solution $\langle \mathcal{E}, C_{\mathcal{E}} \rangle$ of instance $\langle G(V, S), E, \Psi \rangle$, where $\mathcal{E}$ is a planar embedding of G defining the set of facial cycles and $C_{\mathcal{E}} : F(\mathcal{E}) \rightarrow 2^{\{1,\ldots,1(G)\}}$ is an assignment of the connected components of G to the faces of $\mathcal{E}$.

To obtain $\mathcal{E}$, we set the rotation scheme of each vertex v of G in $\mathcal{E}$ to the rotation scheme of v in the embedding $\mathcal{E}_i$ of the component $\mathcal{G}_i$ of the backbone graph G containing v. Clearly, the set of facial cycles $F(\mathcal{E})$ of $\mathcal{E}$ is equal to the union of the set of facial cycles of each $\mathcal{E}_i$, that is, for each face $f \in \mathcal{E}$, we have that f belongs to $\mathcal{E}_i$ for some $1 \leq i \leq 1(\mathcal{G})$.

The assignment function $C_{\mathcal{E}}$ can be defined as follows. Initialize $C_{\mathcal{E}}(f) = \emptyset$, for each facial cycle f in $F(\mathcal{E})$. Then, consider each pair of connected components $\mathcal{G}_i$ and $\mathcal{G}_j$ of the backbone graph and, for each facial cycle f in $F(\mathcal{E}) \cap F(\mathcal{E}_j)$, set $C_{\mathcal{E}}(f) = C_{\mathcal{E}}(f) \cup i$ if $i \in C_{\mathcal{E}_j}(f)$.

We now prove that $(\mathcal{E}, C_{\mathcal{E}})$ is a solution for $\langle G(V, S), E, \Psi \rangle$. Since each $\mathcal{E}_i$ is a planar embedding, then $\mathcal{E}$ is also planar. We just need to prove that for every two faces f' and f'' of $\mathcal{E}$ either (i) $C_{\mathcal{E}}(f') \subseteq C_{\mathcal{E}}(f'')$, or (ii) $C_{\mathcal{E}}(f'') \subseteq C_{\mathcal{E}}(f')$, or (iii) $C_{\mathcal{E}}(f') \cap C_{\mathcal{E}}(f'') = \emptyset$. Clearly, if $f', f'' \in \mathcal{E}_i$ for some i, exactly one of (i), (ii), and (iii) must hold, as otherwise $(\mathcal{E}_i, C_{\mathcal{E}_i})$ would not be a solution of $\langle G(V_i, S_i), E_i, \Psi_i \rangle$. We prove that there exist no $f' \in \mathcal{E}_i$ and $f'' \in \mathcal{E}_j$ with $i \neq j$ such that neither (i), (ii), or (iii) holds. We distinguish three cases according to whether $j \in C_{\mathcal{E}_i}(f')$, or $i \in C_{\mathcal{E}_j}(f'')$, or $j \notin C_{\mathcal{E}_i}(f') \wedge i \notin C_{\mathcal{E}_j}(f'')$. By the connectivity of the union graphs of each instance and by the fact that $(\mathcal{E}_i, C_{\mathcal{E}_i})$ and $(\mathcal{E}_j, C_{\mathcal{E}_j})$ are ω-SDB of $\langle G(V_i, S_i), E_i, \Psi_i \rangle$ and $\langle G(V_j, S_j), E_j, \Psi_j \rangle$, respectively, we have that: (i) must hold, if $i \in C_{\mathcal{E}_j}(f'')$; (ii) must hold, if $j \in C_{\mathcal{E}_i}(f')$; and (iii) must hold,

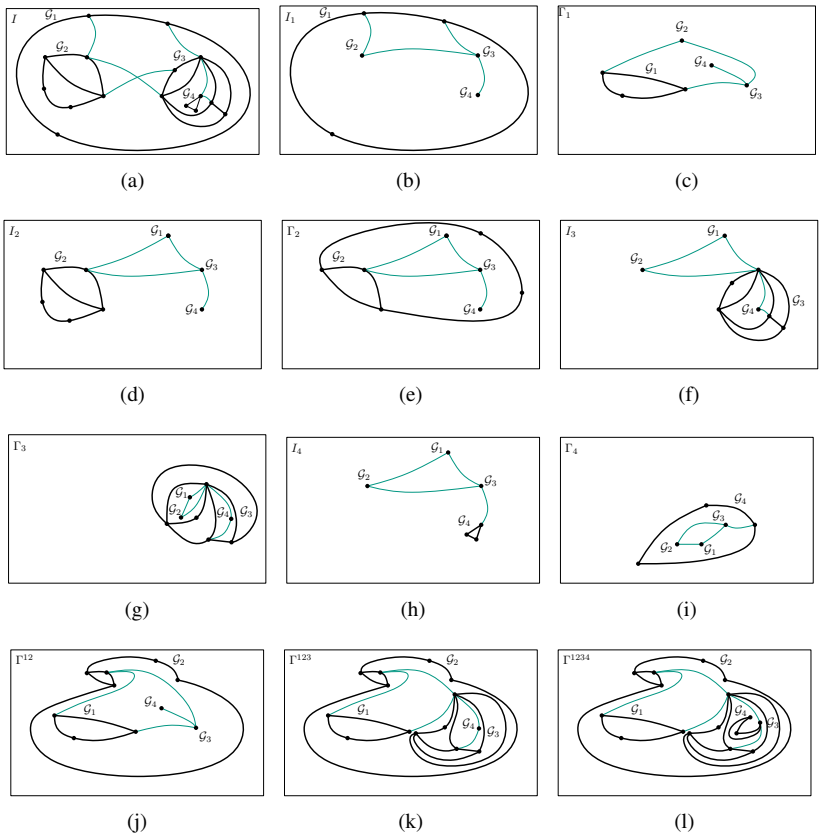


Figure 11.9: (a) Instance $I = \langle G(V,S), E, \Psi \rangle$ of SPB with $\omega = 1$, where G consists of 4 connected components $\mathcal{G}_1$, $\mathcal{G}_2$, $\mathcal{G}_3$, and $\mathcal{G}_4$. Edges of the backbone graph are black thick curves. Edges of the stream are green thin curves. Instances I_1 (b), I_2 (d), I_3 (f), and I_4 (h) obtained by applying the procedure described in the proof of Lemma 11.2 to instance I. 1-SDB Γ_1 (c), Γ_2 (e), I_3 (g), and I_4 (i) of instances I_1, I_2, I_3, and I_4, respectively. (j) 1-SDB Γ^{12} obtained by replacing the drawing of $\mathcal{G}_2$ in Γ_1 with Γ_2. (k) 1-SDB Γ^{123} obtained by replacing the drawing of $\mathcal{G}_3$ in Γ^{12} with Γ^3. (l) 1-SDB Γ^{1234} obtained by replacing the drawing of $\mathcal{G}_4$ in Γ^{123} with Γ_4.

if $j \notin C_{\mathcal{E}_i}(f') \wedge i \notin C_{\mathcal{E}_j}(f'')$. This concludes the proof of the lemma. □

By Lemma 11.2, in the following we only consider the case in which the backbone graph consists of a single non-trivial connected component plus, possibly, isolated vertices. We now present a simple recursive algorithm to test instances with this property.

Algorithm **ALGOCON**.

∘ *INPUT:* an instance $I = \langle G(V,S), E, \Psi \rangle$ of the SPB problem with $\omega = 1$ with union graph $G_\cup$ such that G contains at most one non-trivial connected component.

∘ *OUTPUT:* `YES`, if $\langle G(V,S), E, \Psi \rangle$ is positive, or `NO`, otherwise.

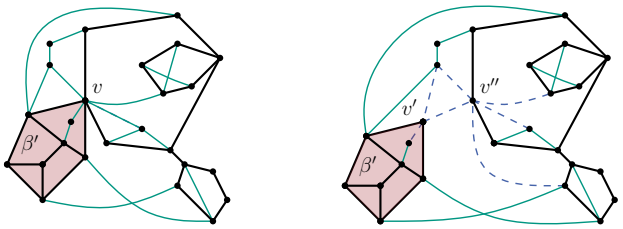
BASE CASE 1: instance I is such that $2(G) = 0$, that is, every connected component of G is an isolated vertex. Return `YES`, as instances of this kind are trivially positive.

BASE CASE 2: instance I is such that (i) $2(G) = 1$, that is, the backbone graph G consists of a single 2-connected component plus, possibly, isolated vertices and (ii) no edge of the stream connects any two isolated vertices. In this case, apply the algorithm of Theorem 11.3 to decide I and return `YES`, if the test succeeds, or `NO`, otherwise.

RECURSIVE STEP: instance I is such that either (`CASE R1`) $2(G) = 1$ and there exists edges of the stream between pairs of isolated vertices or (`CASE R2`) $2(G) > 1$. First, replace instance I with two smaller instances $I^\diamond = \langle G(V_\diamond, S_\diamond), E_\diamond, \Psi_\diamond \rangle$ and $I^\circ = \langle G(V_\circ, S_\circ), E_\circ, \Psi_\circ \rangle$, as described below. Then, return `YES`, if it holds that $ALGOCON(I^\diamond) = ALGOCON(I^\circ) =$ `YES`, or `NO`, otherwise.

CASE R1. Instance $I^\diamond$ is obtained from I by recursively contracting every edge (u,v) of $G_\cup$ with $\{u,v\} \not\subseteq V(\mathcal{G})$. Instance I° is obtained from I by recursively contracting every edge (u,v) of $G_\cup$ with $\{u,v\} \subseteq V(\mathcal{G})$.

CASE R2. Let $\mathcal{G}$ be the unique non-trivial connected component of G, let T be the block-cutvertex tree of $\mathcal{G}$ rooted at any block, and let β be any leaf block in T. Also, let v be the parent cutvertex of β in T. We first construct an auxiliary equivalent instance $I^* = \langle G(V_*, S_*), E_*, \Psi_* \rangle$ starting from I and then obtain instances $I^\diamond$ and I° from I^*, as follows. See Fig. 11.10 for an illustration of the construction of instance I^*. Initialize I^* to I. Replace vertex v in V_* with two vertices v' and v'' and make (i) v' adjacent to all the vertices of β vertex v used to be adjacent to and (ii) v'' adjacent to all the vertices in $V(\mathcal{G}) \setminus V(\beta)$ vertex v used to be adjacent to. Then, replace each edge (v,x) of E^* with edge (v',x), if



(a) (b)

Figure 11.10: (a) Instance I and (b) instance I^* obtained in *CASE R2* of Algorithm ALGOCON. Edges of the backbone graph are black thick curves; edge of the stream are green thin curves; and edges of the stream incident to v' and v'' in I^* are blue dashed curves.

$x \in V(\beta)$ or if $x \in \mathcal{Q}^*$ and there exists a path composed of edges of the stream connecting x to a vertex $y \neq v \in V(\beta)$, and edge (v'', x), if $x \in V(\mathcal{G}) \setminus V(\beta)$ or if $x \in \mathcal{Q}^*$ and there exists a path composed of edges of the stream connecting x to a vertex $y \neq v \in V(\mathcal{G}) \setminus V(\beta)$. Finally, add edge (v', v'') to E^*. It is easy to see that instances I and I^* are equivalent.

Instance $I^\diamond$ is obtained from I^* by recursively contracting every edge (u, v) of $G^*_\cup$ with $u, v \notin V(\beta)$, where $G^*_\cup$ is the union graph of I^*. Instance I° is obtained from I^* by recursively contracting every edge (u, v) of $G^*_\cup$ with $\{u, v\} \subseteq V(\beta)$.

Theorem 11.4 *Let $\langle G(V, S), E, \Psi \rangle$ be an instance of* SPB. *There exists an $O(n+m)$-time algorithm to decide whether $\langle G(V, S), E, \Psi \rangle$ admits an ω-SDB for $\omega = 1$.*

Proof: The algorithm runs in two steps, as follows.

- **STEP 1** applies the reduction illustrated in the proof of Lemma 11.2 to instance $\langle G(V, S), E, \Psi \rangle$ to construct $1(G)$ instances $\langle G(V_i, S_i), E_i, \Psi_i \rangle$ such that the backbone graphs $G(V_i, S_i)$ contain at most one non-trivial connected component.

- **STEP 2** applies Algorithm ALGOCON to every instance $\langle G(V_i, S_i), E_i, \Psi_i \rangle$ and return `YES`, if all such instances are positive, or `NO`, otherwise.

Observe that, the correctness of the presented algorithm follows from the correctness of Lemma 11.2, of Theorem 11.3, and of Algorithm ALGOCON. We now prove the correctness for Algorithm ALGOCON. Obviously, the fact that instances $I^\diamond$ and I° constructed in `CASE R1` and `CASE R2` are both positive is a necessary and sufficient condition for instance I to be positive. We prove termination by induction on the number $2(G)$ of blocks of the backbone graph G of instance I, primarily, and on the number of edges of the stream connecting isolated vertices of the backbone graph, secondarily. (i) If $2(G) = 0$, then `BASE CASE 1` applies and the algorithm stops; (ii) if $2(G) = 1$ and no two isolated vertices of the backbone graph are connected by an edge of the stream, then `BASE CASE 2` applies and the algorithm stops; (iii) if $2(G) = 1$ and there exist edges of the stream between any two isolated vertices of the backbone graph G, then, by `CASE R1`, instance I is split into (a) an instance $I^\diamond$ with $2(G(V_\diamond, E_\diamond)) = 1$ and no edges of the stream connecting any two isolated vertices of the backbone graph $G(V_\diamond, E_\diamond)$, and (b) an instance I° with $2(G(V_\circ, E_\circ)) = 0$; (iv) finally, if $2(G) > 1$, then, by `CASE R2`, instance I is split into (a) an instance I° with $2(G(V_\diamond, E_\diamond)) = 1$ and (b) an instance I° with $2(G(V_\circ, E_\circ)) = 2(G) - 1$.

The running time easily derives from the fact that all instances $\langle G(V_i, S_i), E_i, \Psi_i \rangle$ can be constructed in $O(n+m)$-time and that the algorithm for star instances described in the proof of Theorem 11.3 runs in $O(n + \omega m)$-time. This concludes the proof. □

11.5 Conclusions

In this chapter we introduce the notion of streamed graphs and of ω-stream planarity for such graphs. We studied the STREAMED PLANARITY problem of testing whether a streamed graph is ω-stream planar with respect to a given positive integer window size ω in the two settings with and without backbone graph, settling the question regarding the computational complexity of this problem in both such settings.

On the negative side, we show that there exists a constant ω_0 such that STREAMED PLANARITY with window size ω_0 is NP-complete and that STREAMED PLANARITY WITH BACKBONE is NP-complete for all $\omega \geq 2$ even when the backbone graph is a spanning tree. On the positive side, we provide $O(n + \omega m)$-time algorithms for (i) the case $\omega = 1$ and (ii) all values of ω, if the considered instances are star instances. We remark that, the algorithm for $\omega = 1$ improves to linear time the previously best algorithm for PARTIAL PLANARITY by Schaefer [Sch14] whose running time is $O((nm)^3)$. Moreover, the structural restrictions on the variant for arbitrary values of ω are necessary to overcome the two hardness results and are hence, with this respect, best possible.

Finally, our NP-completeness results rule out, unless P=NP, the existence of FPT algorithms with respect to the window size ω, to the number ξ of exclusive edges, or to the maximum degree θ of the conflict graph for STREAMED PLANARITY (WITH BACKBONE), SEFE, and WEAK REALIZABILITY problems, respectively.

Appendices

Appendix A: Other Research Activities

Simultaneously with the research for the development of this thesis, other topics in the area of Graph Drawing and Network Visualization have been dealt with:

- **Morphing Planar Graph Drawings Optimally**

  A *morph* is a continuous transformation between two topologically equivalent geometric objects. The study of morphs is relevant for several areas of computer science, including computer graphics, animation, and modeling. Many of the geometric shapes that are of interest in these contexts can be effectively described by two-dimensional planar graph drawings. Hence, designing algorithms and establishing bounds for morphing planar graph drawings is an important research challenge. We refer the reader to [GS01, SG01, FE02, EKP03, SG03] for extensive descriptions of the applications of graph drawing morphs.

  It has long been known that there always exists a *planar morph* (that is, a morph that preserves planarity at any time instant) transforming any planar straight-line drawing Γ_s of a plane graph G into any other planar straight-line drawing Γ_t of G. The first proof of such a result, published by Cairns in 1944 [Cai44], was "existential", i.e., no guarantee was provided on the complexity of the trajectories of the vertices during the morph. Almost 40 years later, Thomassen proved in [Tho83] that a morph between Γ_s and Γ_t always exists in which vertices follow trajectories of exponential complexity (in the number of vertices of G). In other words, adopting a setting defined by Grünbaum and Shepard [GS81] which is also the one we consider in this research, Thomassen proved that a sequence $\Gamma_s = \Gamma_1, \Gamma_2, \ldots, \Gamma_k = \Gamma_t$ of planar straight-line drawings of G exists such that, for $1 \leq i \leq k-1$, the *linear morph* transforming Γ_i into Γ_{i+1} is planar, where a linear morph moves each vertex at constant speed along a straight-line trajectory.

A breakthrough was recently obtained by Alamdari *et al.* by proving that a planar morph between any two planar straight-line drawings of the same n-vertex connected plane graph exists in which each vertex follows a trajectory of polynomial complexity [AAC$^+$13]. That is, Alamdari *et al.* showed an algorithm to perform the morph in $O(n^4)$ *morphing steps*, where a morphing step is a linear morph. The $O(n^4)$ bound was shortly afterwards improved to $O(n^2)$ by Angelini *et al.* [AFPR13].

In this research, we provide an algorithm to compute a planar morph with $O(n)$ morphing steps between any two planar straight-line drawings Γ_s and Γ_t of any n-vertex connected plane graph G. Further, we prove that our algorithm is optimal. That is, for every n, there exist two drawings Γ_s and Γ_t of the same n-vertex plane graph (in fact a path) such that any planar morph between Γ_s and Γ_t consists of $\Omega(n)$ morphing steps. To the best of our knowledge, no super-constant lower bound was previously known.

The schema of our algorithm is the same as in [AAC$^+$13, AFPR13]. Namely, we morph Γ_s and Γ_t into two drawings Γ_s^x and Γ_t^x in which a certain vertex v can be contracted onto a neighbor x. Such contractions generate two straight-line planar drawings Γ'_s and Γ'_t of a smaller plane graph G'. A morph between Γ'_s and Γ'_t is recursively computed and suitably modified to produce a morph between Γ_s and Γ_t. The main ingredient for our new bound is a drastically improved algorithm to morph Γ_s and Γ_t into Γ_s^x and Γ_t^x. In fact, while the task of making v contractible onto x is accomplished with $O(n)$ morphing steps in [AAC$^+$13, AFPR13], we devise and use properties of monotone drawings, level planar drawings, and hierarchical graphs to perform it with $O(1)$ morphing steps. The algorithm can be extended to work for disconnected graphs at the expense of an increase in the number of steps to $O(n^{1.5})$ [ABC$^+$15].

The idea behind the lower bound is that linear morphs can poorly simulate rotations, that is, a morphing step rotates an edge of an angle whose size is $O(1)$. We then consider two drawings Γ_s and Γ_t of an n-vertex path P, where Γ_s lies on a straight-line, whereas Γ_t has a spiral-like shape, and we prove that in any planar morph between Γ_s and Γ_t there is one edge of P whose total rotation describes an angle whose size is $\Omega(n)$.

This research is a joint work with Patrizio Angelini, Giuseppe Di Battista, Fabrizio Frati, Maurizio Patrignani, and Vincenzo Roselli, appeared in [ALD$^+$14a].

- **Optimal Morphs of Convex Drawings**

  Convex drawings of plane graphs are a classical topic of investigation in geometric graph theory. A characterization [Tho84] of the plane graphs that admit convex drawings and a linear-time algorithm [CYN84] to test whether a graph admits a convex drawing are known. Convex drawings in small area [KLTT97, BR06, BFM07], orthogonal convex drawings [RNN98, RNG04, Tho84], and convex drawings satisfying a variety of further geometric constraints [HN10a, HN10b] have also been studied. It is intuitive, but far from trivial to prove, that the space of the convex drawings of any n-vertex plane graph G is connected; i.e., the points in $\mathbb{R}^{2n}$, each corresponding to the two-dimensional coordinates of a convex drawing of G, form a connected set. Expressed in yet another way, there exists a *convex morph* between any two convex drawings Γ_s and Γ_t of the same plane graph G, that is, a continuous deformation from Γ_s to Γ_t so that the intermediate drawing of G is convex at any instant of the deformation. The main result of this research is the existence of a convex morph between any two convex drawings of the same plane graph such that each vertex moves along a piecewise linear curve with linear complexity during the deformation.

  The existence of a convex morph between any two convex drawings was first proved by Thomassen [Tho83] more than 30 years ago. Thomassen's result confirmed a conjecture of Grünbaum and Shepard [GS81] and improved upon a previous result of Cairns [Cai44], stating that a continuous deformation, called *morph*, between any two straight-line planar drawings of the same plane graph G exists such that any intermediate straight-line drawing of G is planar. More recently, motivated by applications in computer graphics, animation, and modeling, a number of algorithms for morphing graph drawings have been designed [EKP03, FE02, GS01, SG01, SG03]. These algorithms aim to construct morphs that preserve the topology of the given drawings at any time, while guaranteeing that the trajectories of the vertices are "nice" curves.

  In this research we give an algorithm to construct a convex morph between any two convex drawings of the same n-vertex plane graph with $O(n)$ morphing steps. Our algorithm preserves the convexity of the drawing at any time instant and in fact preserves strict convexity, if the given drawings are strictly-convex. The linear bound is tight in the worst case, as can be shown by adapting the lower bound construction of Angelini *et al.* [ALD$^+$14a]. We remark that Thomassen's algorithm [Tho83] constructs convex morphs with an exponential number of steps. To the best of our knowledge, no other algorithm is known to construct a convex morph between any two convex drawings of the same plane graph.

The outline of our algorithm is simple. Let Γ_s and Γ_t be two convex drawings of the same *convex graph* G, that is, a plane graph that admits a convex drawing. Determine a connected subgraph G' of G such that removing G' from G results in a smaller convex graph G''. Then G' lies inside one face f of G''. Morph Γ_s into a drawing Γ'_s of G and morph Γ_t into a drawing Γ'_t of G such that the cycle of G corresponding to f is delimited by a convex polygon in Γ'_s and in Γ'_t. These morphs consist of one morphing step each. Remove G' from Γ'_s and Γ'_t to obtain two convex drawings Γ''_s and Γ''_t of G''. Finally, recursively compute a morph between Γ''_s and Γ''_t. Since f remains convex throughout the whole morph from Γ''_s to Γ''_t, a morph of G from Γ'_s to Γ'_t can be obtained from the morph of G'' from Γ''_s to Γ''_t by suitably drawing G' inside f at each intermediate step of such a morph. The final morph from Γ_s to Γ_t consists of the morph from Γ_s to Γ'_s followed by the morph from Γ'_s to Γ'_t, and then the reverse of the morph from Γ_t to Γ'_t. Our algorithm has two main ingredients.

The first ingredient is a structural decomposition of convex graphs that generalizes a well-known structural decomposition of triconnected planar graphs due to Grünbaum and Shepard [GS81]. The latter states that any subdivision of a triconnected planar graph contains a path whose removal results in a subdivision of a smaller triconnected planar graph. For convex graphs we can prove a similar theorem which states, roughly speaking, that any convex graph contains a path, or three paths incident to the same vertex, whose removal results in a smaller convex graph. Our approach is thus based on *removing* a subgraph from the input graph. This differs from the recent papers on morphing graph drawings [AAC$^+$13, ALD$^+$14a], where the basic operation is to *contract* a vertex to a neighbor (i.e. move it arbitrarily close to a neighbor). One of the difficulties of the previous approach was to determine a trajectory for a contracted vertex inside the moving polygon of its neighbors. By removing a subgraph and forcing the newly formed face to be convex, we avoid this difficulty.

The second ingredient is a relationship between *unidirectional morphs* and level planar drawings of hierarchical graphs, which allows us to compute the above mentioned morphs between Γ_s and Γ'_s and between Γ_t and Γ'_t with one morphing step. This relationship was first observed by Angelini *et al.* [ALD$^+$14a]. However, in order to use it in our setting, we need to prove that every strictly-convex graph admits a *strictly-convex* level planar drawing; this strengthens a result of Hong and Nagamochi [HN10a] and might be of independent interest.

We leave open the question whether any two convex drawings of the same plane graph G can be morphed so that every intermediate drawing has polynomial *size* (e.g., the ratio between the length of any two edges is polynomial in the size of

G during the entire morph). For the purpose of solving this problem positively, our approach seems to be better than previous ones; intuitively, subgraph removals are more suitable than vertex contractions for a morphing algorithm that doesn't blow up the size of the intermediate drawings. Nevertheless, we haven't yet been able to prove that polynomial-size morphs always exist.

This research is a joint work with Patrizio Angelini, Fabrizio Frati, Anna Lubiw, Maurizio Patrignani, and Vincenzo Roselli, appeared in [ADF+15].

- **Anchored Drawings of Planar Graphs**

  Several applications require to draw graphs whose vertices are constrained to be not too much *distant* from specific points [LMR98, AAnS05]. As an example, consider a graph whose vertices are cities and whose edges are relationships between cities. It is conceivable that the user wants to draw the graph on a geographic map where vertices have the coordinates of the corresponding cities. Unfortunately, depending on the local density of the cities, the drawing may be cluttered or may contain crossings between edges that might disappear if the vertices could move from their locations. Hence, the user may be interested to trade precision for quality of the drawing, accepting that the vertices move of a certain distance from the location of the cities, provided that the readability of the drawing increases. Problems in which the input consists of a set of imprecise points have also been studied in Computational Geometry [DM03, LvK10].

  In this work we consider the following problem, that we call ANCHORED GRAPH DRAWING (AGD)[3]. Given a graph $G = (V, E)$, an initial placement for its vertices, and a distance δ, we ask whether there exists a planar drawing of G, according to a certain drawing convention, such that each vertex $v \in V$ can move at distance at most δ from its initial placement. Note that the problem can have different formulations depending on how the concepts of "readability" and "distance" are defined.

  We consider both straight-line planar drawings and rectilinear planar drawings. Further, in addition to the traditional L_2 Euclidean distance, we consider the L_1 Manhattan distance and the L_∞ 'uniform' distance. Note that, adopting L_2 distance is equivalent to allowing vertices to be placed into circular regions centered at their original positions, and adopting L_1 or L_∞ distances is equivalent to allowing vertices to be placed into diamond-shaped or square-shaped areas, respectively.

[3] We remark that the term 'anchored graph' was used within a different setting in [CM13].

Metric	Distance	Region Shape	Straight-line	Rectilinear
L_1	Manhattan	$\diamond$	NP-hard	NP-hard
L_2	Euclidean	$\bigcirc$	NP-hard	NP-hard
L_∞	Uniform	$\square$	NP-hard	Polynomial

Table A.1: The complexity of the ANCHORED GRAPH DRAWING problem depending on the metric and drawing style adopted when the areas of the vertices do not overlap.

Observe that, if the regions of two vertices overlap, the positions of the two vertices can be swapped with respect to their initial placement, which may be confusing to a user of the drawing. Moreover, overlapping between vertex regions would make problem AGD as difficult as known Clustered Planarity variants, such as the Strip Planarity problem [ADDF13a] (see also Chapter 6) in the straight-line setting, whose complexity is a non-trivial open problem. Hence, we restrict to instances such that the regions of the vertices do not overlap.

We remark that the version of the problem where each circle may have a different size was shown to be NP-hard in [God95] by reducing Planar-$(3, 4)$-SAT with variable repetitions (where repeated occurrences of one variable in one clause are counted repeatedly). The proof in [God95] uses disks with radius zero and disks with large radii. Also, the reduction relies on overlapping disks.

Furthermore, we observe that the NP-hardness of the problem with different distances and overlapping areas trivially follows from the NP-hardness of extending a planar straight-line drawing [Pat06] by setting $\delta(v) = 0$ for each fixed vertex v and allowing suitably large distances for vertices that have to be planarly added to the drawing.

In this work we show that the ANCHORED GRAPH DRAWING problem is NP-hard for any combination of metrics and drawing standards that we considered, with the exception of rectilinear drawings and uniform distance metric (square-shaped regions). These results, summarized in Table A.1, were somehow unexpected, as computing a planar rectilinear drawing of a graph, without any further constraint, is NP-hard [GT01b]. We leave open the following questions: 1. Does problem AGD belong to class NP? 2. The instances in our NP-hardness proofs can be augmented to equivalent instances whose graphs are biconnected (we omit details for space reasons). In these instances, different truth values correspond to different embeddings. What is the complexity of AGD when the input graph is triconnected or has at least a fixed embedding? 3. What if we

allow the vertex regions to (at least partially) overlap?

This research is a joint work with Patrizio Angelini, Marco Di Bartolomeo, Giuseppe Di Battista, Seok-Hee Hong, Maurizio Patrignani, and Vincenzo Roselli, appeared in [ADD+14].

- **Drawing Georeferenced Graphs: Combining Graph Drawing and Geographic Data**

  In this research, we address the problem of exploring a *georeferenced graph*, i.e., a graph with some geographic information associated with its nodes. Namely, we consider the task of visually exploring relationships (such as established connections, similarity, reachability, etc) among a set of *georeferenced entities*, i.e., entities that have geographic data associated with them. A novel 2.5D paradigm, briefly described in the following, is proposed that provides a robust and practical solution based on separating and then integrating back again the networked and geographical dimensions of the input dataset. This allows us to easily cope with partial or incomplete geographic annotations, to reduce cluttering of close entities, and to address focus-plus-context visualization issues. Typical application domains include, for example, coordinating search and rescue teams or medical evacuation squads, monitoring ad-hoc networks, exploring location-based social networks and, more in general, visualizing relational datasets including geographic annotations.

  The purpose of the interface is to represent in the most intuitive and unambiguous way both the relationships among the entities and their positions, conveying at the same time the degree of uncertainty associated with the geographic information. We propose an innovative 2.5D paradigm to visually explore data with both a relational and a geolocalized nature. Our strategy is to separate and simultaneously visualize the networked and the geographic information of the input dataset. Namely, the geographic information is represented on the *geographic layer*, which is in the bottom part of the interface, while the networked information is represented on the *logical layer*, which is parallel to the geographic layer and placed in the upper part of the interface. Leaders among the two layers relate nodes with their geographic locations, if any. The interface is shown in Fig. A.1. In order to avoid overlaps between the two layers, which would give occlusion among the two types of information, we restrict their size to two equally-sized rectangles and suitably place the point of view on the longest side of the rectangles.

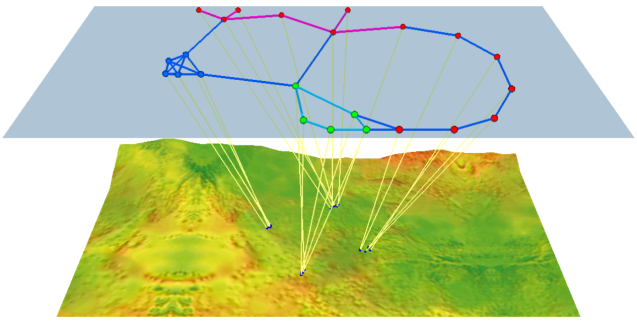


Figure A.1: A snapshot of the proposed 2.5D interface taken from a JavaScript demonstrative prototype, implemented using the WebGL graphics library [The13], that runs within any compatible web browser. The *logical layer*, above, shows the networked data, while the *geographic layer*, below, displays actual locations of the entities contained in the logical layer, whenever available.

Nodes are placed on the logical layer with the purpose of conveying as effectively as possible the structure of the graph, reducing cluttering and crossings among edges. To this purpose, we devised a specialized force-directed algorithm, called Algorithm `retina`, that computes the network layout directly on the logical layer. The obtained layout tries to achieve both evenly-spaced distribution of nodes and few crossings among edges, while seeking to minimize the distance of each node from the corresponding position on the geographic layer.

The purpose of the geographic layer, instead, is that of displaying the current position of each node shown on the logical layer. Such a position is represented by means of a marker on the map with a straight-line leader connecting it to the corresponding node on the logical layer. When the position of the node is affected by uncertainty the marker on the map is a geometric shape, usually a circle, enclosing the area where the corresponding object is supposed to be, and the leader consists of a cone with its apex on the node. Nodes with no geographic information associated have no marker on the map.

Therefore, in our approach, we have two types of links: (i) the edges on the logical layer and (ii) the leaders connecting the two layers. We privilege the readability of the graph induced by the first type of links, by trying to reduce crossings on the logical layer, which severely jeopardize the comprehension of the graph structure [PCJ97, Pur00]. Crossings among leaders have lower impact

on readability, since the structure of the graph induced by them is of limited interest to the user. In fact, each leader establishes a connection between an entity and a geographic location, and paths of leaders are never considered by the user. Also, when the mouse hovers a vertex on the logical layer, the interface highlights its incident leader to help the user identify the leader endpoint on the geographic layer.

This research is a joint work with Marco Di Bartolomeo, Maurizio Patrignani, Giuseppe Di Battista, Davide Cannone, and Sergio Tortora, appeared in [DDP+15].

- **Drawing Graphs on a Smartphone**

  Millions of people in the world have in their pocket a smartphone and such a number is rapidly increasing [Gar10]. Such a widely used device is exploited to quickly access, from almost everywhere, different types of data on different subjects and a large amount of such data is relational information. For example, smartphones are used to access social networks like Facebook or Twitter, ontologies like the Wikipedia network of concepts, or technical information related to the job of the smartphone's owner like the connections of a computer network or the delivery routes of a product distribution system.

  Graph Drawing can play an important role in supporting information visualization on the smartphones, provided that the methodologies and the tools that are typical of this research area are recast to meet the needs of such a challenging device. Indeed, different information contexts have already changed their visualization methods in this direction. For instance, on-line newspapers have special visualization formats that are designed for the smartphone. However, as far as we know, the only previous attempt to draw graphs on smartphones [Pix08] uses traditional Graph Drawing techniques.

  Dealing with smartphones, the main challenge that visualization applications have to face is, of course, the small screen size. On the other hand, such a strong limitation comes together with new technological opportunities that can be exploited to support the interaction. They are the multi-touch screen that is able to capture commonly used gestures like pinch, flick, and slide, sensors like the accelerometer and the compass, and sounds and vibrations.

  Since any graph is too large for the little screen of the smartphones, a pivotal reference point for designing interfaces and algorithms for drawing graphs on such a device is the literature on drawing very large graphs. One, for example,

could use the fish-eye approach [SB92] where the details of the drawing decrease according to the distance that separates them from a point chosen by the user. However, using Shneiderman's information visualization mantra [Shn96] (overview first, zoom and filter, then details-on-demand) in this context seems to be unfeasible. In fact, it is unclear how to provide, on such a small screen, a suitable overview of the information.

In this work we present a system for the visualization and navigation of relational information on the smartphones. We devised a visualization and interaction paradigm that (i) is based on showing to the user only a small subgraph defined by a focus vertex and its neighborhood and (ii) exploits smartphone-specific interaction primitives to explore the graph. The paradigm is inspired by the navigation approach of [ECH97]. Even such a simple visualization paradigm can originate interesting algorithmic issues and requires non-trivial Graph Drawing algorithms to be implemented. We conducted an extensive experimentation that put in evidence the effectiveness of the designed algorithms and exploit our system to visualize data coming from challenging domains. Namely, we show several customizations of the system aimed at exploring and at visualizing popular Web contents like social networks and Wikipedia. Current implementations include the iPhone and the Google Android platforms. Several problems related to our paradigm are left open: 1. How to apply variations of our paradigm for graph visualization on handheld devices, like the iPad, whose screen has a size larger that a smartphone but smaller than a usual computer screen? Variations of the paradigms could include: the visualization of more than one focus vertex, the placement of lobe vertices on several concentric layers, or the usage of all the sides of the screen. 2. A typical way to visualize large graphs it to use clustering techniques. Is it possible to combine our paradigm with existing techniques for the visualization of clustered graphs? 3. Can the heuristics that we have presented for selecting a lobe be improved both from the point of view of the performance and from the point of view of the effectiveness?

This research is a joint work with Giuseppe Di Battista and Francesco Ingrassia appeared in [DDI10] and published in a journal [DDI12].

Appendix B: List of Publications

Journal Publications

1. Patrizio Angelini, Giordano Da Lozzo, Giuseppe Di Battista, Fabrizio Frati, Vincenzo Roselli. The Importance of Being Proper (In Clustered-Level Planarity and T-Level Planarity). *Theoretical Computer Science*. 571:1-9. 2015.

2. Patrizio Angelini, Giordano Da Lozzo, Daniel Neuwirth. Advancements on SEFE and Partitioned Book Embedding Problems. *Theoretical Computer Science*. 575:71-89. 2015.

3. Patrizio Angelini, Giordano Da Lozzo, Giuseppe Di Battista, Fabrizio Frati, Maurizio Patrignani, Vincenzo Roselli. Relaxing the Constraints of Clustered Planarity. *Computational Geometry: Theory and Applications*. 48(2):42-75. 2015.

4. Giordano Da Lozzo, Giuseppe Di Battista, Claudio Squarcella. Visual Discovery of the Correlation between BGP Routing and Round-Trip Delay Active Measurements. *Computing*. 96(1): 67-77. 2014.

5. Giordano Da Lozzo, Giuseppe Di Battista, Francesco Ingrassia. Drawing Graphs on a Smartphone. *Journal of Graph Algorithms and Applications, Special Issue of Selected Papers from GD '10*. 16(1):109-126. 2012.

Conference Publications

1. Patrizio Angelini, Giordano Da Lozzo, Fabrizio Frati, Anna Lubiw, Maurizio Patrignani, Vincenzo Roselli. Optimal Morphs of Convex Drawings. *In Proc. 31st Symposium on Computational Geometry (SoCG '15)*, 2015. To appear.

2. Giordano Da Lozzo, Ignaz Rutter. Planarity of Streamed Graphs. *In Proc. 9th International Conference on Algorithms and Complexity (CIAC '15)*, 2015. To appear.

3. Giordano Da Lozzo, Marco Di Bartolomeo, Maurizio Patrignani, Giuseppe Di Battista, Davide Cannone, Sergio Tortora. Drawing Georeferenced Graphs - Combining Graph Drawing and Geographic Data. *In Proc. 6th International Conference on Information Visualization Theory and Applications (IVAPP '15), pages 109-116, 2015.*

4. Giordano Da Lozzo, Vìt Jelìnek, Jan Kratochvì, Ignaz Rutter. Planar Embeddings with Small and Uniform Faces. *In Proc. 25th International Symposium on Algorithms and Computation (ISAAC '14)*, volume 8889, pages 633-645, 2014.

5. Patrizio Angelini, Giordano Da Lozzo, Giuseppe Di Battista, Fabrizio Frati, Vincenzo Roselli. The Importance of Being Proper (In Clustered-Level Planarity and T-Level Planarity). *In Proc. 22nd International Symposium on Graph Drawing (GD '14)*, volume 8871, pages 246-258, 2014.

6. Patrizio Angelini, Giordano Da Lozzo, Marco Di Bartolomeo, Giuseppe Di Battista, Seok-Hee Hong, Maurizio Patrignani, Vincenzo Roselli. Anchored Drawings of Planar Graphs. *In Proc. 22nd International Symposium on Graph Drawing (GD '14)*, volume 8871, pages 404-415, 2014.

7. Patrizio Angelini, Giordano Da Lozzo, Giuseppe Di Battista, Fabrizio Frati, Maurizio Patrignani, Vincenzo Roselli. Morphing Planar Graph Drawings Optimally. *In Proc. 41st International Colloquium on Automata, Languages and Programming (ICALP '14)*, volume 8572, pages 126-137, 2014.

8. Patrizio Angelini, Giordano Da Lozzo, Daniel Neuwirth. On some NP-complete SEFE Problems. *In Proc. Workshop on Algorithms and Computation (WALCOM '14)*, volume 8344, pages 193-205, 2014.

9. Patrizio Angelini, Giordano Da Lozzo, Giuseppe Di Battista, Fabrizio Frati. Strip Planarity Testing. *In Proc. 21st International Symposium on Graph Drawing (GD '13)*, volume 8242, pages 37-48, 2013.

10. Patrizio Angelini, Carla Binucci, Giordano Da Lozzo, Walter Didimo, Luca Grilli, Fabrizio Montecchiani, Maurizio Patrignani, Ioannis Tollis. Drawing Non-planar Graphs with Crossing-free Subgraphs. *In Proc. 21st International Symposium on Graph Drawing (GD '13)*, volume 8242, pages 295-307, 2013.

11. Giordano Da Lozzo, Giuseppe Di Battista, Francesco Ingrassia. Drawing Graphs on a Smartphone. *In Proc. 18th International Symposium on Graph Drawing (GD '10)*, volume 6502, pages 153-164, 2010.

Technical Reports

1. Giordano Da Lozzo, Ignaz Rutter. Planarity of Streamed Graphs. *Tech. Rep. arXiv:1501.07106, Cornell University*, 2014.

2. Giordano Da Lozzo, Vìt Jelìnek, Jan Kratochvì, Ignaz Rutter. Planar Embeddings with Small and Uniform Faces. *Tech. Rep. arXiv:1409.4299, Cornell University*, 2014.

3. Patrizio Angelini, Giordano Da Lozzo, Giuseppe Di Battista, Fabrizio Frati, Vincenzo Roselli. On the Complexity of Clustered-Level Planarity and T-Level Planarity. *Tech. Rep. arXiv:1406.6533, Cornell University*, 2014.

4. Patrizio Angelini, Giordano Da Lozzo. Deepening the Relationship between SEFE and C-Planarity. *Tech. Rep. arXiv:1404.6175, Cornell University*, 2014.

5. Patrizio Angelini, Giordano Da Lozzo, Giuseppe Di Battista, Fabrizio Frati, Maurizio Patrignani, Vincenzo Roselli. Morphing Planar Graph Drawings Optimally. *Tech. Rep. arXiv:1402.4364, Cornell University*, 2014.

6. Patrizio Angelini, Giordano Da Lozzo, Daniel Neuwirth. Advancements on SEFE and Partitioned Book Embedding Problems. *Tech. Rep. arXiv:1311.3607, Cornell University*, 2014.

7. Patrizio Angelini, Carla Binucci, Giordano Da Lozzo, Walter Didimo, Luca Grilli, Fabrizio Montecchiani and Maurizio Patrignani, Ioannis Tollis. Algorithms and Bounds for Drawing Non-planar Graphs with Crossing-free Subgraphs. *Tech. Rep. arXiv:1308.6706, Cornell University*, 2013.

8. Patrizio Angelini, Giordano Da Lozzo and Giuseppe Di Battista, Fabrizio Frati. Strip Planarity Testing of Embedded Planar Graphs. *Tech. Rep. arXiv:1309.0683, Cornell University*, 2013.

9. Patrizio Angelini, Giordano Da Lozzo, Giuseppe Di Battista, Fabrizio Frati, Maurizio Patrignani, Vincenzo Roselli. Relaxing the Constraints of Clustered Planarity. *Tech. Rep. arXiv:1207.3934, Cornell University*, 2012.

Others

1. Patrizio Angelini, Giordano Da Lozzo. SEFE = C-Planarity?. *In Booklet 9th International Colloquium on Graph Theory and Combinatorics (ICGT '14)*, 2014.

2. Giordano Da Lozzo, Giuseppe Di Battista, Claudio Squarcella. Visual Discovery of the Correlation between BGP Routing and Round-Trip Delay Active Measurements. *In 1st IMC Workshop on Internet Visualization (WIV '12)*, 2012.

Bibliography

[AAC+13] Soroush Alamdari, Patrizio Angelini, Timothy M. Chan, Giuseppe Di Battista, Fabrizio Frati, Anna Lubiw, Maurizio Patrignani, Vincenzo Roselli, Sahil Singla, and Bryan T. Wilkinson. Morphing planar graph drawings with a polynomial number of steps. In *Proceedings of the Twenty-Fourth Annual ACM-SIAM Symposium on Discrete Algorithms, SODA 2013, New Orleans, Louisiana, USA, January 6-8, 2013*, pages 1656–1667, 2013.

[AAnS05] Manuel Abellanas, Andrés Aiello, Gregorio Hernández Pe nalver, and Rodrigo I. Silveira. Network drawing with geographical constraints on vertices. In *Actas XI Encuentros de Geom. Comput.*, pages 111–118, 2005.

[ABC+15] Greg Aloupis, Luis Barba, Paz Carmi, Vida Dujmovic, Fabrizio Frati, and Pat Morin. Compatible connectivity-augmentation of planar disconnected graphs. In Piotr Indyk, editor, *Proceedings of the Twenty-Sixth Annual ACM-SIAM Symposium on Discrete Algorithms, SODA 2015, San Diego, CA, USA, January 4-6, 2015*, pages 1602–1615, 2015.

[ABD+13] Patrizio Angelini, Carla Binucci, Giordano Da Lozzo, Walter Didimo, Luca Grilli, Fabrizio Montecchiani, Maurizio Patrignani, and Ioannis G. Tollis. Drawing non-planar graphs with crossing-free subgraphs. In Stephen K. Wismath and Alexander Wolff, editors, *Graph Drawing - 21st International Symposium, GD 2013, Bordeaux, France, September 23-25, 2013, Revised Selected Papers*, LNCS, pages 292–303, 2013.

[ACDP13] Patrizio Angelini, Pier Francesco Cortese, Giuseppe Di Battista, and Maurizio Patrignani. Topological morphing of planar graphs. *Theor. Comput. Sci.*, 514:2–20, 2013.

[Ack09] Eyal Ackerman. On the maximum number of edges in topological graphs with no four pairwise crossing edges. *Discrete & Computational Geometry*, 41(3):365–375, 2009.

[AD14] Patrizio Angelini and Giordano Da Lozzo. SEFE = C-Planarity?, 2014.

[ADD12] Patrizio Angelini, Marco Di Bartolomeo, and Giuseppe Di Battista. Implementing a partitioned 2-page book embedding testing algorithm. In *Graph Drawing*, pages 79–89, 2012.

[ADD+14] Patrizio Angelini, Giordano Da Lozzo, Marco Di Bartolomeo, Giuseppe Di Battista, Seok-Hee Hong, Maurizio Patrignani, and Vincenzo Roselli. Anchored drawings of planar graphs. In Christian A. Duncan and Antonios Symvonis, editors, *Graph Drawing - 22nd International Symposium, GD 2014, Würzburg, Germany, September 24-26, 2014, Revised Selected Papers*, volume 8871 of *LNCS*, pages 404–415. Springer, 2014.

[ADD+15] Patrizio Angelini, Giordano Da Lozzo, Giuseppe Di Battista, Fabrizio Frati, Maurizio Patrignani, and Vincenzo Roselli. Relaxing the constraints of clustered planarity. *Computational Geometry: Theory and Applications*, 48(2):42–75, 2015.

[ADDF13a] Patrizio Angelini, Giordano Da Lozzo, Giuseppe Di Battista, and Fabrizio Frati. Strip planarity testing. In *Graph Drawing - 21st International Symposium, GD 2013, Bordeaux, France, September 23-25, 2013, Revised Selected Papers*, pages 37–48, 2013.

[ADDF13b] Patrizio Angelini, Giordano Da Lozzo, Giuseppe Di Battista, and Fabrizio Frati. Strip planarity testing of embedded planar graphs. Tech. Report arXiv:1309.0683, Cornell University, 2013.

[ADF+10] Patrizio Angelini, Giuseppe Di Battista, Fabrizio Frati, Vít Jelínek, Jan Kratochvíl, Maurizio Patrignani, and Ignaz Rutter. Testing planarity of partially embedded graphs. In Moses Charikar, editor, *Proceedings of the Twenty-First Annual ACM-SIAM Symposium on Discrete Algorithms, SODA 2010, Austin, Texas, USA, January 17-19, 2010*, pages 202–221, 2010.

[ADF+12] Patrizio Angelini, Giuseppe Di Battista, Fabrizio Frati, Maurizio Patrignani, and Ignaz Rutter. Testing the simultaneous embeddability of two graphs whose intersection is a biconnected or a connected graph. *J. Discrete Algorithms*, 14:150–172, 2012.

[ADF13] Patrizio Angelini, Giuseppe Di Battista, and Fabrizio Frati. Simultaneous embedding of embedded planar graphs. *Int. J. Comput. Geometry Appl.*, 23(2):93–126, 2013.

[ADF+15] Patrizio Angelini, Giordano Da Lozzo, Fabrizio Frati, Anna Lubiw, Maurizio Patrignani, and Vincenzo Roselli. Optimal morphs of convex drawings. In *Symposium on Computational Geometry 2015, SoCG '15, Eindhoven, the Netherlands, June 22-25, 2015*, 2015. To appear.

[ADN14] Patrizio Angelini, Giordano Da Lozzo, and Daniel Neuwirth. On some NP-complete SEFE problems. In Sudebkumar Prasant Pal and Kunihiko Sadakane, editors, *WALCOM*, volume 8344 of *LNCS*, pages 200–212. Springer, 2014.

[ADN15] Patrizio Angelini, Giordano Da Lozzo, and Daniel Neuwirth. Advancements on SEFE and partitioned book embedding problems. *Theoretical Computer Science*, 575:71–89, 2015.

[ADP11] Patrizio Angelini, Giuseppe Di Battista, and Maurizio Patrignani. Finding a minimum-depth embedding of a planar graph in $O(n^4)$ time. *Algorithmica*, 60:890–937, 2011.

[AFG10] Patrizio Angelini, Fabrizio Frati, and Luca Grilli. An algorithm to construct greedy drawings of triangulations. *J. Graph Algorithms Appl.*, 14(1):19–51, 2010.

[AFK11] Patrizio Angelini, Fabrizio Frati, and Michael Kaufmann. Straight-line rectangular drawings of clustered graphs. *Discrete & Computational Geometry*, 45(1):88–140, 2011.

[AFP09] Patrizio Angelini, Fabrizio Frati, and Maurizio Patrignani. Splitting clusters to get c-planarity. In David Eppstein and Emden R. Gansner, editors, *Graph Drawing, 17th International Symposium, GD 2009, Chicago, IL, USA, September 22-25, 2009. Revised Papers*, pages 57–68, 2009.

[AFPR13] Patrizio Angelini, Fabrizio Frati, Maurizio Patrignani, and Vincenzo Roselli. Morphing planar graph drawings efficiently. In Stephen K. Wismath and Alexander Wolff, editors, *Graph Drawing - 21st International Symposium, GD 2013, Bordeaux, France, September 23-25, 2013, Revised Selected Papers*, pages 49–60, 2013.

[AHR10] Sarmad Abbasi, Patrick Healy, and Aimal Rextin. Improving the running time of embedded upward planarity testing. *Inf. Process. Lett.*, 110(7):274–278, 2010.

[AHU83] Alfred V. Aho, John E. Hopcroft, and Jeffrey D. Ullman. *Data Structures and Algorithms*. Addison-Wesley, 1983.

[ALD+14a] Patrizio Angelini, Giordano Da Lozzo, Giuseppe Di Battista, Fabrizio Frati, Maurizio Patrignani, and Vincenzo Roselli. Morphing planar graph drawings optimally. In Javier Esparza, Pierre Fraigniaud, Thore Husfeldt, and Elias Koutsoupias, editors, *Automata, Languages, and Programming - 41st International Colloquium, ICALP 2014, Copenhagen, Denmark, July 8-11, 2014, Proceedings, Part I*, volume 8572 of *LNCS*, pages 126–137. Springer, 2014.

[ALD+14b] Patrizio Angelini, Giordano Da Lozzo, Giuseppe Di Battista, Fabrizio Frati, and Vincenzo Roselli. The importance of being proper - (in clustered-level planarity and t-level planarity). In Christian A. Duncan and Antonios Symvonis, editors, *Graph Drawing - 22nd International Symposium, GD 2014, Würzburg, Germany, September 24-26, 2014, Revised Selected Papers*, pages 246–258, 2014.

[ALD+15] Patrizio Angelini, Giordano Da Lozzo, Giuseppe Di Battista, Fabrizio Frati, and Vincenzo Roselli. The importance of being proper (in clustered-level planarity and t-level planarity). *Theor. Comput. Sci.*, 571:1–9, 2015.

[AP89] Stefan Arnborg and Andrzej Proskurowski. Linear time algorithms for NP-hard problems restricted to partial k-trees. *Discrete Applied Mathematics*, 23(1):11–24, 1989.

[AT07] Eyal Ackerman and Gábor Tardos. On the maximum number of edges in quasi-planar graphs. *J. Comb. Theory, Ser. A*, 114(3):563–571, 2007.

[BBF05] Christian Bachmaier, Franz-Josef Brandenburg, and Michael Forster. Radial level planarity testing and embedding in linear time. *J. Graph Algorithms Appl.*, 9(1):53–97, 2005.

[BCD+07] Peter Braß, Eowyn Cenek, Christian A. Duncan, Alon Efrat, Cesim Erten, Dan Ismailescu, Stephen G. Kobourov, Anna Lubiw, and Joseph S. B. Mitchell. On simultaneous planar graph embeddings. *Comput. Geom.*, 36(2):117–130, 2007.

[BCG+13] Christoph Buchheim, Markus Chimani, Carsten Gutwenger, Michael Jünger, and Petra Mutzel. Crossings and planarization. In Roberto Tamassia, editor, *Handbook of Graph Drawing and Visualization*. CRC Press, 2013.

[BDD00] Paola Bertolazzi, Giuseppe Di Battista, and Walter Didimo. Computing orthogonal drawings with the minimum number of bends. *IEEE Trans. Computers*, 49(8):826–840, 2000.

[BDD+12] Carla Binucci Ulrik Brandes, Giuseppe Di Battista, Walter Didimo, Marco Gaertler, Pietro Palladino, Maurizio Patrignani, Antonios Symvonis, and Katharina Anna Zweig. Drawing trees in a streaming model. *Inf. Process. Lett.*, 112(11):418–422, 2012.

[BDLM94] Paola Bertolazzi, Giuseppe Di Battista, Giuseppe Liotta, and Carlo Mannino. Upward drawings of triconnected digraphs. *Algorithmica*, 12(6):476–497, 1994.

[BEG+12] Franz-Josef Brandenburg, David Eppstein, Andreas Gleißner, Michael T. Goodrich, Kathrin Hanauer, and Josef Reislhuber. On the density of maximal 1-planar graphs. In *GD*, volume 7704 of *LNCS*, pages 327–338. Springer, 2012.

[BFM07] Nicolas Bonichon, Stefan Felsner, and Mohamed Mosbah. Convex drawings of 3-connected plane graphs. *Algorithmica*, 47(4):399–420, 2007.

[BKM98] Therese C. Biedl, Michael Kaufmann, and Petra Mutzel. Drawing planar partitions II: hh-drawings. In Juraj Hromkovic and Ondrej Sýkora, editors, *Graph-Theoretic Concepts in Computer Science, 24th International Workshop, WG '98, Smolenice Castle, Slovak Republic, June 18-20, 1998, Proceedings*, pages 124–136, 1998.

[BKR13a] Thomas Bläsius, Annette Karrer, and Ignaz Rutter. Simultaneous embedding: Edge orderings, relative positions, cutvertices. In Stephen K. Wismath and Alexander Wolff, editors, *Graph Drawing - 21st International Symposium, GD 2013, Bordeaux, France, September 23-25, 2013, Revised Selected Papers*, pages 220–231, 2013.

[BKR13b] Thomas Bläsius, Stephen G. Kobourov, and Ignaz Rutter. Simultaneous embedding of planar graphs. In R. Tamassia, editor, *Handbook of Graph Drawing and Visualization*. CRC Press, 2013.

[BKRW14] Thomas Bläsius, Marcus Krug, Ignaz Rutter, and Dorothea Wagner. Orthogonal graph drawing with flexibility constraints. *Algorithmica*, 68:859–885, 2014.

[BL76] Kellogg S. Booth and George S. Lueker. Testing for the consecutive ones property, interval graphs, and graph planarity using pq-tree algorithms. *J. Comput. Syst. Sci.*, 13(3):335–379, 1976.

[BM76] John A. Bondy and Uppaluri S. R Murty. *Graph theory with applications*. American Elsevier Publishing Co., Inc., New York, 1976.

[BM88] Daniel Bienstock and Clyde L. Monma. On the complexity of covering vertices by faces in a planar graph. *SIAM J. Comput.*, 17(1):53–76, 1988.

[BM90] Daniel Bienstock and Clyde L. Monma. On the complexity of embedding planar graphs to minimize certain distance measures. *Algorithmica*, 5(1):93–109, 1990.

[BM04] John M. Boyer and Wendy J. Myrvold. On the cutting edge: simplified $O(n)$ planarity by edge addition. *Journal of Graph Algorithms and Applications*, 8(3):241–273, 2004.

[Boo75] Kellogg S. Booth. *PQ-tree algorithms*. PhD thesis, University of California, Berkeley, 1975.

[BR06] Imre Bárány and Günter Rote. Strictly convex drawings of planar graphs. *Documenta Math.*, 11:369–391, 2006.

[BR13] Thomas Bläsius and Ignaz Rutter. Simultaneous PQ-ordering with applications to constrained embedding problems. In *SODA*, pages 1030–1043, 2013.

[BR14] Thomas Bläsius and Ignaz Rutter. A new perspective on clustered planarity as a combinatorial embedding problem. In Christian A. Duncan and Antonios Symvonis, editors, *Graph Drawing - 22nd International Symposium, GD 2014, Würzburg, Germany, September 24-26, 2014, Revised Selected Papers*, pages 440–451, 2014.

[BRW13] Thomas Bläsius, Ignaz Rutter, and Dorothea Wagner. Optimal orthogonal graph drawing with convex bend costs. In Fedor V. Fomin, Rūsiņš Freivalds, Marta Kwiatkowsak, and David Peleg, editors, *Automata, Languages, and Programming (ICALP'13)*, volume 7965 of *LNCS*, pages 184–195. Springer, 2013.

[Cai44] S. S. Cairns. Deformations of plane rectilinear complexes. *American Math. Monthly*, 51:247–252, 1944.

[CB05] Pier Francesco Cortese and Giuseppe Di Battista. Clustered planarity. In Joseph S. B. Mitchell and Günter Rote, editors, *Proceedings of the 21st ACM Symposium on Computational Geometry, Pisa, Italy, June 6-8, 2005*, pages 32–34, 2005.

[CDF+08] Pier Francesco Cortese, Giuseppe Di Battista, Fabrizio Frati, Maurizio Patrignani, and Maurizio Pizzonia. C-planarity of c-connected clustered graphs. *J. Graph Algorithms Appl.*, 12(2):225–262, 2008.

[CDFK14] Markus Chimani, Giuseppe Di Battista, Fabrizio Frati, and Karsten Klein. Advances on testing c-planarity of embedded flat clustered graphs. In Christian A. Duncan and Antonios Symvonis, editors, *Graph Drawing - 22nd International Symposium, GD 2014, Würzburg, Germany, September 24-26, 2014, Revised Selected Papers*, pages 416–427, 2014.

[CDPP05] Pier Francesco Cortese, Giuseppe Di Battista, Maurizio Patrignani, and Maurizio Pizzonia. Clustering cycles into cycles of clusters. *J. Graph Algorithms Appl.*, 9(3):391–413, 2005.

[CDPP09] Pier Francesco Cortese, Giuseppe Di Battista, Maurizio Patrignani, and Maurizio Pizzonia. On embedding a cycle in a plane graph. *Discrete Mathematics*, 309(7):1856–1869, 2009.

[CEGL12] Erin W. Chambers, David Eppstein, Michael T. Goodrich, and Maarten Löffler. Drawing graphs in the plane with a prescribed outer face and polynomial area. *J. Graph Algorithms Appl.*, 16(2):243–259, 2012.

[CEX15] Hsien-Chih Chang, Jeff Erickson, and Chao Xu. Detecting weakly simple polygons. In *Proceedings of the Twenty-Sixth Annual ACM-SIAM Symposium on Discrete Algorithms, SODA 2015, San Diego, CA, USA, January 4-6, 2015*, pages 1655–1670. SIAM, 2015.

[Cho34] Ch. Chojnacki. Über wesentlich unplättbare kurven im dreidimensionalen raume. *Fundamenta Mathematicae*, 23(1):135–142, 1934.

[CLRS09] Thomas H. Cormen, Charles E. Leiserson, Ronald L. Rivest, and Clifford Stein. *Introduction to Algorithms*. MIT Press, 2009.

[CM13] Sergio Cabello and Bojan Mohar. Adding one edge to planar graphs makes crossing number and 1-planarity hard. *SIAM J. Comput.*, 42(5):1803–1829, 2013.

[CN88] Norishige Chiba and Takao Nishizeki. *Planar Graphs: Theory and Algorithms*. Annals of Discrete Mathematics 32. North-Holland, Amsterdam, 1988.

[CN98] Marek Chrobak and Shin-Ichi Nakano. Minimum-width grid drawings of plane graphs. *Computational Geometry*, 11(1):29 – 54, 1998.

[CR05] Derek G. Corneil and Udi Rotics. On the relationship between clique-width and treewidth. *SIAM Journal on Computing*, 34(4):825–847, 2005.

[CW06] Sabine Cornelsen and Dorothea Wagner. Completely connected clustered graphs. *J. Discrete Algorithms*, 4(2):313–323, 2006.

[CYN84] Norishige Chiba, Toru Yamanouchi, and Takao Nishizeki. Linear algorithms for convex drawings of planar graphs. In J. A. Bondy and U. S. R. Murty, editors, *Progress in Graph Theory*, pages 153–173. Academic Press, New York, NY, 1984.

[Dah98] Elias Dahlhaus. A linear time algorithm to recognize clustered graphs and its parallelization. In *LATIN '98: Theoretical Informatics, Third Latin American Symposium, Campinas, Brazil, April, 20-24, 1998, Proceedings*, LNCS, pages 239–248. Springer, 1998.

[dCvO08] Mark de Berg, Otfried Cheong, Marc van Kreveld, and Mark Overmars. *Computational Geometry: Algorithms and Applications*. Springer-Verlag TELOS, Santa Clara, CA, USA, 3rd edition, 2008.

[DDI10] Giordano Da Lozzo, Giuseppe Di Battista, and Francesco Ingrassia. Drawing graphs on a smartphone. In Ulrik Brandes and Sabine Cornelsen, editors, *Graph Drawing - 18th International Symposium, GD 2010, Konstanz, Germany, September 21-24, 2010. Revised Selected Papers*, LNCS, pages 153–164. Springer, 2010.

[DDI12] Giordano Da Lozzo, Giuseppe Di Battista, and Francesco Ingrassia. Drawing graphs on a smartphone. *J. Graph Algorithms Appl.*, 16(1):109–126, 2012.

[DDLM05] Emilio Di Giacomo, Walter Didimo, Giuseppe Liotta, and Henk Meijer. Computing radial drawings on the minimum number of circles. *J. Graph Algorithms Appl.*, 9(3):365–389, 2005.

[DDLM13] Emilio Di Giacomo, Walter Didimo, Giuseppe Liotta, and Fabrizio Montecchiani. Area requirement of graph drawings with few crossings per edge. *Computational Geometry*, 46(8):909–916, 2013.

[DDM01] Giuseppe Di Battista, Walter Didimo, and A. Marcandalli. Planarization of clustered graphs. In Petra Mutzel, Michael Jünger, and Sebastian Leipert, editors, *Graph Drawing, 9th International Symposium, GD 2001 Vienna, Austria, September 23-26, 2001, Revised Papers*, pages 60–74, 2001.

[DDP+15] Giordano Da Lozzo, Marco Di Bartolomeo, Maurizio Patrignani, Giuseppe Di Battista, Davide Cannone, and Sergio Tortora. Drawing georeferenced graphs - combining graph drawing and geographic data. In Lars Linsen, Andreas Kerren, and José Braz, editors, *Proceedings of the 6th International Conference on Information Visualization Theory and Applications, IVAPP 2015, Berlin, Germany, 11-14 March, 2015.*, pages 109–116, 2015.

[DEL11] Walter Didimo, Peter Eades, and Giuseppe Liotta. Drawing graphs with right angle crossings. *Theoretical Computer Science*, 412(39):5156–5166, 2011.

[DETT99] Giuseppe Di Battista, Peter Eades, Roberto Tamassia, and Ioannis G. Tollis. *Graph Drawing: Algorithms for the Visualization of Graphs*. Prentice-Hall, 1999.

[DF09] Giuseppe Di Battista and Fabrizio Frati. Efficient c-planarity testing for embedded flat clustered graphs with small faces. *J. Graph Algorithms Appl.*, 13(3):349–378, 2009.

[dFOdM12] Hubert de Fraysseix and Patrice Ossona de Mendez. Trémaux trees and planarity. *European Journal of Combinatorics*, 33(3):279–293, April 2012.

[DGK03] Christian A. Duncan, Michael T. Goodrich, and Stephen G. Kobourov. Planarity-preserving clustering and embedding for large planar graphs. *Comput. Geom.*, 24(2):95–114, 2003.

[Did13] Walter Didimo. Density of straight-line 1-planar graph drawings. *Information Processing Leters*, 113(7):236–240, 2013.

[Die05] Reinhard Diestel. *Graph theory*. Graduate texts in mathematics. Springer, Berlin, 2005.

[DL12] Walter Didimo and Giuseppe Liotta. The crossing angle resolution in graph drawing. In János Pach, editor, *Thirty Essays on Geometric Graph Theory*. Springer, 2012.

[DLV98] Giuseppe Di Battista, Giuseppe Liotta, and Francesco Vargiu. Spirality and optimal orthogonal drawings. *SIAM Journal on Computing*, 27(6):1764–1811, 1998.

[DM03] Adrian Dumitrescu and Joseph S. B. Mitchell. Approximation algorithms for TSP with neighborhoods in the plane. *J. Algorithms*, 48(1):135–159, 2003.

[dPP88] Hubert de Fraysseix, Janos Pach, and Richard Pollack. Small Sets Supporting Fáry Embeddings of Planar Graphs. In Janos Simon, editor, *Symposium on Theory of Computing (STOC '88)*, pages 426–433, 1988.

[dPP90] Hubert de Fraysseix, Janos Pach, and Richard Pollack. How to draw a planar graph on a grid. *Combinatorica*, 10(1):41–51, 1990.

[dR82] Hubert de Fraysseix and Pierre Rosenstiehl. A depth-first-search characterization of planarity. *Annals of Discrete Mathematics*, 13:75–80, 1982.

[DR15] Giordano Da Lozzo and Ignaz Rutter. Planarity of streamed graphs. In *Proc. 9th International Conference on Algorithms and Complexity (CIAC '15)*, LNCS, 2015. To appear.

[DT88] Giuseppe Di Battista and Roberto Tamassia. Algorithms for plane representations of acyclic digraphs. *Theor. Comput. Sci.*, 61:175–198, 1988.

[DT90] Giuseppe Di Battista and Roberto Tamassia. On-line graph algorithms with SPQR-trees. In Mike Paterson, editor, *Automata, Languages and Programming, 17th International Colloquium, ICALP90, Warwick University, England, July 16-20, 1990, Proceedings*, LNCS, pages 598–611. Springer, 1990.

[DT96a] Giuseppe Di Battista and Roberto Tamassia. On-line maintenance of triconnected components with spqr-trees. *Algorithmica*, 15(4):302–318, 1996.

[DT96b] Giuseppe Di Battista and Roberto Tamassia. On-line planarity testing. *SIAM J. Comput.*, 25(5):956–997, 1996.

[ECH97] Peter Eades, Robert F. Cohen, and Mao Lin Huang. Online animated graph drawing for web navigation. In G. Di Battista, editor, *Proc. 5th Int. Symp. Graph Drawing, GD*, number 1353 in LNCS, pages 330–335. Springer-Verlag, 1997.

[Ede87] Herbert Edelsbrunner. *Algorithms in Combinatorial Geometry*, volume 10 of *EATCS Monographs on Theoretical Computer Science*. Springer, 1987.

[EFK09] Alejandro Estrella-Balderrama, J. Joseph Fowler, and Stephen G. Kobourov. On the characterization of level planar trees by minimal patterns. In David Eppstein and Emden R. Gansner, editors, *Graph Drawing, 17th International Symposium, GD 2009, Chicago, IL, USA, September 22-25, 2009. Revised Papers*, pages 69–80, 2009.

[EFLN06] Peter Eades, Qing-Wen Feng, Xuemin Lin, and Hiroshi Nagamochi. Straight-line drawing algorithms for hierarchical graphs and clustered graphs. *Algorithmica*, 44(1):1–32, 2006.

[EHK+12] Peter Eades, Seok-Hee Hong, Naoki Katoh, Giuseppe Liotta, Pascal Schweitzer, and Yusuke Suzuki. Testing maximal 1-planarity of graphs with a rotation system in linear time - (extended abstract). In *GD*, volume 7704 of *LNCS*, pages 339–345. Springer, 2012.

[EK05] Cesim Erten and Stephen G. Kobourov. Simultaneous embedding of planar graphs with few bends. *J. Graph Algorithms Appl.*, 9(3):347–364, 2005.

[EKLN05] Cesim Erten, Stephen G. Kobourov, Vu Le, and Armand Navabi. Simultaneous graph drawing: Layout algorithms and visualization schemes. *J. Graph Algorithms Appl.*, 9(1):165–182, 2005.

[EKP03] Cesim Erten, Stephen G. Kobourov, and Chandan Pitta. Intersection-free morphing of planar graphs. In Giuseppe Liotta, editor, *Graph Drawing, 11th International Symposium, GD 2003, Perugia, Italy, September 21-24, 2003, Revised Papers*, pages 320–331, 2003.

[ET76] Shimon Even and Robert E. Tarjan. Computing an *st*-numbering. *Theoretical Computer Science*, 2:339–344, 1976.

[Eve79] Shimon Even. *Graph Algorithms*. W. H. Freeman & Co, New York, USA, 1979.

[Fár48] István Fáry. On straight line representation of planar graphs. *Acta Universitaria Szegediensis, Sectio Scientiarum Mathematicarum*, 11:229–233, 1948.

[FB04] Michael Forster and Christian Bachmaier. Clustered level planarity. In Peter van Emde Boas, Jaroslav Pokorný, Mária Bieliková, and Julius Stuller, editors, *SOFSEM*, volume 2932 of *LNCS*, pages 218–228, 2004.

[FCE95a] Qing-Wen Feng, Robert F. Cohen, and Peter Eades. How to draw a planar clustered graph. In Ding-Zhu Du and Ming Li, editors, *Computing and Combinatorics, First Annual International Conference, COCOON '95, Xi'an, China, August 24-26, 1995, Proceedings*, pages 21–30, 1995.

[FCE95b] Qing-Wen Feng, Robert F. Cohen, and Peter Eades. Planarity for clustered graphs. In Paul G. Spirakis, editor, *Algorithms - ESA '95, Third Annual European Symposium, Corfu, Greece, September 25-27, 1995, Proceedings*, pages 213–226, 1995.

[FE02] Carsten Friedrich and Peter Eades. Graph drawing in motion. *J. Graph Algorithms Appl.*, 6(3):353–370, 2002.

[Fen97] Qing-Wen Feng. *Algorithms for Drawing Clustered Graphs*. PhD thesis, Department of Computer Science and Software engineering, University of Newclastle, April 1997.

[FGJ+08] J. Joseph Fowler, Carsten Gutwenger, Michael Jünger, Petra Mutzel, and Michael Schulz. An SPQR-tree approach to decide special cases of simultaneous embedding with fixed edges. In Ioannis G. Tollis and Maurizio Patrignani, editors, *Graph Drawing, 16th International Symposium, GD 2008, Heraklion, Crete, Greece, September 21-24, 2008. Revised Papers*, volume 5417 of *LNCS*, pages 157–168. Springer, 2008.

[FK07] J. Joseph Fowler and Stephen G. Kobourov. Minimum level nonplanar patterns for trees. In Seok-Hee Hong, Takao Nishizeki, and Wu Quan, editors, *Graph Drawing, 15th International Symposium, GD 2007, Sydney, Australia, September 24-26, 2007. Revised Papers*, pages 69–75, 2007.

[FKMP95] Michael R. Fellows, Jan Kratochvíl, Martin Middendorf, and Frank Pfeiffer. The complexity of induced minors and related problems. *Algorithmica*, 13:266–282, 1995.

[For05] Michael Forster. *Crossings in clustered level graphs*. PhD thesis, University of Passau, 2005.

[FP08] Fabrizio Frati and Maurizio Patrignani. A note on minimum area straight-line drawings of planar graphs. In Seok-Hee Hong, Takao Nishizeki, and Wu Quan, editors, *Graph Drawing*, volume 4875 of *LNCS*, pages 339–344. Springer Berlin Heidelberg, 2008.

[FPS13] Jacob Fox, János Pach, and Andrew Suk. The number of edges in k-quasi-planar graphs. *SIAM Journal on Discrete Mathematics*, 27(1):550–561, 2013.

[Ful14] Radoslav Fulek. Towards the hanani-tutte theorem for clustered graphs. In Dieter Kratsch and Ioan Todinca, editors, *Graph-Theoretic Concepts in Computer Science - 40th International Workshop, WG 2014, Nouan-le-Fuzelier, France, June 25-27, 2014. Revised Selected Papers*, pages 176–188, 2014.

[Gab83] Harold N. Gabow. An efficient reduction technique for degree-constrained subgraph and bidirected network flow problems. In *Theory of Computing (STOC'83)*, pages 448–456. ACM, 1983.

[Gar10] Gartner. Press releases. http://www.gartner.com/, 2010.

[GDLM12] Emilio Di Giacomo, Walter Didimo, Giuseppe Liotta, and Fabrizio Montecchiani. h-quasi planar drawings of bounded treewidth graphs in linear area. In Martin Charles Golumbic, Michal Stern, Avivit Levy, and Gila Morgenstern, editors, *Graph-Theoretic Concepts in Computer Science - 38th International Workshop, WG 2012, Jerusalem, Israel, June 26-28, 2012, Revised Selcted Papers*, pages 91–102, 2012.

[GJ77] M. R. Garey and David S. Johnson. The rectilinear steiner tree problem in NP-complete. *SIAM Journal of Applied Mathematics*, 32:826–834, 1977.

[GJ83] M. R. Garey and D. S. Johnson. Crossing Number is NP-Complete. *SIAM Journal on Algebraic and Discrete Methods*, 4(3):312–316, 1983.

[GJL+02] Carsten Gutwenger, Michael Jünger, Sebastian Leipert, Petra Mutzel, Merijam Percan, and René Weiskircher. Advances in c-planarity testing of clustered graphs. In Stephen G. Kobourov and Michael T. Goodrich, editors, *Graph Drawing, 10th International Symposium, GD 2002, Irvine, CA, USA, August 26-28, 2002, Revised Papers*, pages 220–235, 2002.

[GJL+03] Carsten Gutwenger, Michael Jünger, Sebastian Leipert, Petra Mutzel, Merijam Percan, and René Weiskircher. Subgraph induced planar connectivity augmentation: (extended abstract). In *Graph-Theoretic Concepts in Computer Science, 29th International Workshop, WG 2003, Elspeet, The Netherlands, June 19-21, 2003, Revised Papers*, volume 2880 of *LNCS*, pages 261–272. Springer, 2003.

[GJP+06] Elisabeth Gassner, Michael Jünger, Merijam Percan, Marcus Schaefer, and Michael Schulz. Simultaneous graph embeddings with fixed edges. In Fedor V. Fomin, editor, *Graph-Theoretic Concepts in Computer Science, 32nd International Workshop, WG 2006, Bergen, Norway, June 22-24, 2006, Revised Papers*, pages 325–335, 2006.

[GKM08] Carsten Gutwenger, Karsten Klein, and Petra Mutzel. Planarity testing and optimal edge insertion with embedding constraints. *J. Graph Algorithms Appl.*, 12(1):73–95, 2008.

[GLS05] Michael T. Goodrich, George S. Lueker, and Jonathan Z. Sun. C-planarity of extrovert clustered graphs. In Patrick Healy and Nikola S. Nikolov, editors, *Graph Drawing, 13th International Symposium, GD 2005, Limerick, Ireland, September 12-14, 2005, Revised Papers*, pages 211–222, 2005.

[GM00] Carsten Gutwenger and Petra Mutzel. A linear time implementation of spqr-trees. In Joe Marks, editor, *Graph Drawing, 8th International Symposium, GD 2000, Colonial Williamsburg, VA, USA, September 20-23, 2000, Proceedings*, pages 77–90, 2000.

[GM04] Carsten Gutwenger and Petra Mutzel. Graph embedding with minimum depth and maximum external face (extended abstract). In Giuseppe Liotta, editor, *Graph Drawing (GD'03)*, volume 2912 of *LNCS*, pages 259–272. Springer, 2004.

[GMW05] Carsten Gutwenger, Petra Mutzel, and René Weiskircher. Inserting an edge into a planar graph. *Algorithmica*, 41(4):289–308, 2005.

[God95] Michael Godau. On the difficulty of embedding planar graphs with inaccuracies. In Roberto Tamassia and Ioannis Tollis, editors, *Graph Drawing (GD '94)*, volume 894 of *LNCS*, pages 254–261. Springer, 1995.

[GS81] B. Grunbaum and G.C. Shephard. *The geometry of planar graphs*. Cambridge University Press, 1981.

[GS01] Craig Gotsman and Vitaly Surazhsky. Guaranteed intersection-free polygon morphing. *Computers & Graphics*, 25(1):67–75, 2001.

[GT01a] Ashim Garg and Roberto Tamassia. On the computational complexity of upward and rectilinear planarity testing. *SIAM J. Comput.*, 31(2):601–625, 2001.

[GT01b] Ashim Garg and Roberto Tamassia. On the computational complexity of upward and rectilinear planarity testing. *SIAM J. Comput.*, 31(2):601–625, 2001.

[GT09] Michael T. Goodrich and Roberto Tamassia. *Algorithm Design: Foundations, Analysis and Internet Examples*. John Wiley & Sons, Inc., New York, NY, USA, 2nd edition, 2009.

[Har69] Frank Harary. *Graph theory*. Addison-Wesley, 1969.

[HELP12] Seok-Hee Hong, Peter Eades, Giuseppe Liotta, and Sheung-Hung Poon. Fáry's theorem for 1-planar graphs. In *COCOON*, volume 7434 of *LNCS*, pages 335–346. Springer, 2012.

[HHE08] Weidong Huang, Seok-Hee Hong, and Peter Eades. Effects of crossing angles. In *PacificVis*, pages 41–46. IEEE, 2008.

[HJL13] Bernhard Haeupler, Krishnam Raju Jampani, and Anna Lubiw. Testing simultaneous planarity when the common graph is 2-connected. *J. Graph Algorithms Appl.*, 17(3):147–171, 2013.

[HKL04] Patrick Healy, Ago Kuusik, and Sebastian Leipert. A characterization of level planar graphs. *Discrete Mathematics*, 280(1-3):51–63, 2004.

[HL96] Michael D. Hutton and Anna Lubiw. Upward planning of single-source acyclic digraphs. *SIAM J. Comput.*, 25(2):291–311, 1996.

[HN09] Seok-Hee Hong and Hiroshi Nagamochi. Two-page book embedding and clustered graph planarity. Technical report, Dept. of Applied Mathematics and Physics, University of Kyoto, Japan, 2009.

[HN10a] Seok-Hee Hong and Hiroshi Nagamochi. Convex drawings of hierarchical planar graphs and clustered planar graphs. *J. Discrete Algorithms*, 8(3):282–295, 2010.

[HN10b] Seok-Hee Hong and Hiroshi Nagamochi. A linear-time algorithm for symmetric convex drawings of internally triconnected plane graphs. *Algorithmica*, 58(2):433–460, 2010.

[HN14] Seok-Hee Hong and Hiroshi Nagamochi. Simpler algorithms for testing two-page book embedding of partitioned graphs. In Zhipeng Cai, Alex Zelikovsky, and Anu G. Bourgeois, editors, *COCOON '14*, volume 8591 of *LNCS*, pages 477–488. Springer, 2014.

[Hos12] Daniel Hoske. *Book embedding with fixed page assignments*. Bachelor thesis, Karlsruhe Institute of Technology, Karlsruhe, Germany, 2012.

[HP66] Frank Harary and Geert Prins. The block-cutpoint-tree of a graph. *Publicationes Mathematicae Debrecen*, 13:103–107, 1966.

[HT73] John E. Hopcroft and Robert Endre Tarjan. Efficient algorithms for graph manipulation [H] (algorithm 447). *Commun. ACM*, 16(6):372–378, 1973.

[HT74] John E. Hopcroft and Robert Endre Tarjan. Efficient planarity testing. *J. ACM*, 21(4):549–568, 1974.

[HT08] Bernhard Haeupler and Robert E. Tarjan. Planarity algorithms via PQ-trees (extended abstract). *Electronic Notes in Discrete Mathematics*, 31:143–149, 2008.

[JJKL08] Vít Jelínek, Eva Jelínková, Jan Kratochvíl, and Bernard Lidický. Clustered planarity: Embedded clustered graphs with two-component clusters. In Ioannis G. Tollis and Maurizio Patrignani, editors, *Graph Drawing, 16th International Symposium, GD 2008, Heraklion, Crete, Greece, September 21-24, 2008. Revised Papers*, pages 121–132, 2008.

[JKK+07] Eva Jelínková, Jan Kára, Jan Kratochvíl, Martin Pergel, Ondrej Suchý, and Tomás Vyskocil. Clustered planarity: Small clusters in eulerian graphs. In Seok-Hee Hong, Takao Nishizeki, and Wu Quan, editors, *Graph Drawing, 15th International Symposium, GD 2007, Sydney, Australia, September 24-26, 2007. Revised Papers*, pages 303–314, 2007.

[JKK+09] Eva Jelínková, Jan Kára, Jan Kratochvíl, Martin Pergel, Ondrej Suchý, and Tomás Vyskocil. Clustered planarity: Small clusters in cycles and eulerian graphs. *J. Graph Algorithms Appl.*, 13(3):379–422, 2009.

[JKR13] Vít Jelínek, Jan Kratochvíl, and Ignaz Rutter. A Kuratowski-type theorem for planarity of partially embedded graphs. *Comput. Geom.*, 46(4):466–492, 2013.

[JL02] Michael Jünger and Sebastian Leipert. Level planar embedding in linear time. *J. Graph Algorithms Appl.*, 6(1):67–113, 2002.

[JLM98] Michael Jünger, Sebastian Leipert, and Petra Mutzel. Level planarity testing in linear time. In Sue Whitesides, editor, *Graph Drawing, 6th International Symposium, GD'98, Montréal, Canada, August 1998, Proceedings*, pages 224–237, 1998.

[JS09] Michael Jünger and Michael Schulz. Intersection graphs in simultaneous embedding with fixed edges. *J. of Graph Algorithms and Applications*, 13(2):205–218, 2009.

[JSTV08] Vít Jelínek, Ondrej Suchý, Marek Tesar, and Tomás Vyskocil. Clustered planarity: Clusters with few outgoing edges. In Ioannis G. Tollis and Maurizio Patrignani, editors, *Graph Drawing, 16th International Symposium, GD 2008, Heraklion, Crete, Greece, September 21-24, 2008. Revised Papers*, pages 102–113, 2008.

[JW93] Klaus Jansen and Gerhard J. Woeginger. The complexity of detecting crossingfree configurations in the plane. *BIT Numerical Mathematics*, 33(4):580–595, 1993.

[Kam06] Frank Kammer. Simultaneous embedding with two bends per edge in polynomial area. In Lars Arge and Rusins Freivalds, editors, *Algorithm Theory - SWAT 2006, 10th ScandinavianWorkshop on Algorithm Theory, Riga, Latvia, July 6-8, 2006, Proceedings*, pages 255–267, 2006.

[Kan96] Goos Kant. Drawing planar graphs using the canonical ordering. *Algorithmica*, 16(1):4–32, 1996.

[Kir88] David G. Kirkpatrick. Establishing order in planar subdivisions. *Discrete & Computational Geometry*, 3:267–280, 1988.

[KK03] Lukasz Kowalik and Maciej Kurowski. Short path queries in planar graphs in constant time. In Lawrence L. Larmore and Michel X. Goemans, editors, *Proceedings of the 35th Annual ACM Symposium on Theory of Computing, June 9-11, 2003, San Diego, CA, USA*, pages 143–148, 2003.

[KLN91] Jan Kratochvíl, Anna Lubiw, and Jaroslav Nesetril. Noncrossing subgraphs in topological layouts. *SIAM J. on Discrete Mathematics*, 4(2):223–244, 1991.

[KLTT97] Goos Kant, Giuseppe Liotta, Roberto Tamassia, and Ioannis G. Tollis. Area requirement of visibility representations of trees. *Inf. Process. Lett.*, 62(2):81–88, 1997.

[KM13] Vladimir P. Korzhik and Bojan Mohar. Minimal obstructions for 1-immersions and hardness of 1-planarity testing. *Journal of Graph Theory*, 72(1):30–71, 2013.

[Kra98] Jan Kratochvíl. Crossing number of abstract topological graphs. In Sue Whitesides, editor, *GD*, volume 1547 of *LNCS*, pages 238–245. Springer, 1998.

[KSSW07] Christian Knauer, Étienne Schramm, Andreas Spillner, and Alexander Wolff. Configurations with few crossings in topological graphs. *Computational Geometry*, 37(2):104–114, 2007.

[Kur30] Kazimierz Kuratowski. Sur le problème des courbes gauches en topologie. *Fundamenta Mathematicae*, 15:271–283, 1930.

[KW01] Michael Kaufmann and Dorothea Wagner, editors. *Drawing Graphs: methods and models*. LNCS. Springer, Berlin, New York, 2001.

[Len89] Thomas Lengauer. Hierarchical planarity testing algorithms. *J. ACM*, 36(3):474–509, 1989.

[LJKR14] Giordano Da Lozzo, Vít Jelínek, Jan Kratochvíl, and Ignaz Rutter. Planar embeddings with small and uniform faces. In *Algorithms and Computation - 25th International Symposium, ISAAC 2014, Jeonju, Korea, December 15-17, 2014, Proceedings*, volume 8889 of *LNCS*, pages 633–645. Springer, 2014.

[LMR98] Kelly A. Lyons, Henk Meijer, and David Rappaport. Algorithms for cluster busting in anchored graph drawing. *J. Graph Algorithms Appl.*, 2(1), 1998.

[LPW05] Gad M. Landau, Laxmi Parida, and Oren Weimann. Gene proximity analysis across whole genomes via PQ trees[1]. *Journal of Computational Biology*, 12(10):1289–1306, 2005.

[LvK10] Maarten Löffler and Marc J. van Kreveld. Largest and smallest convex hulls for imprecise points. *Algorithmica*, 56(2):235–269, 2010.

[MM11] Cristopher Moore and Stephan Mertens. *The Nature of Computation*. Oxford University Press, 2011.

[MR01] Christopher Moore and John M. Robson. Hard tiling problems with simple tiles. *Discrete Comput. Geom.*, 26(4):573–590, 2001.

[MW99] Petra Mutzel and René Weiskircher. Optimizing over all combinatorial embeddings of a planar graph (extended abstract). In Gérard Cornuéjols, Rainer E. Burkard, and Gerhard J. Woeginger, editors, *Integer Programming and Combinatorial Optimization (IPCO'99)*, volume 1610 of *LNCS*, pages 361–376. Springer, 1999.

[NR04] Takao Nishizeki and Md. Saidur Rahman. *Planar Graph Drawing*, volume 12 of *Lecture Notes Series on Computing*. World Scientific, Singapore, 2004.

[Opa79] Jaroslav Opatrny. Total ordering problem. *SIAM J. Comput.*, 8(1):111–114, 1979.

[Pat06] Maurizio Patrignani. On extending a partial straight-line drawing. *International Journal of Foundations of Computer Science (IJFCS)*, 17(5):1061–1069, 2006.

[PCA02] Helen C. Purchase, David A. Carrington, and Jo-Anne Allder. Empirical evaluation of aesthetics-based graph layout. *Empirical Software Engineering*, 7(3):233–255, 2002.

[PCJ97] Helen C. Purchase, Robert F. Cohen, and Murray I. James. An experimental study of the basis for graph drawing algorithms. *ACM Journal of Experimental Algorithmics*, 2:4, 1997.

[Pix08] Pixelglow. Instaviz. http://instaviz.com/, 2008.

[PS85] Franco P. Preparata and Michael I. Shamos. *Computational Geometry: An Introduction*. Springer, 1985.

[PSS96] János Pach, Farhad Shahrokhi, and Mario Szegedy. Applications of the crossing number. *Algorithmica*, 16(1):111–117, 1996.

[PT97] János Pach and Géza Tóth. Graphs drawn with few crossings per edge. *Combinatorica*, 17(3):427–439, 1997.

[PT00a] János Pach and Géza Tóth. Which crossing number is it anyway? *J. Comb. Theory, Ser. B*, 80(2):225–246, 2000.

[PT00b] Maurizio Pizzonia and Roberto Tamassia. Minimum depth graph embedding. In Mike Paterson, editor, *Algorithms - ESA 2000, 8th Annual European Symposium, Saarbrücken, Germany, September 5-8, 2000, Proceedings*, LNCS, pages 356–367. Springer, 2000.

[Pur00] Helen C. Purchase. Effective information visualisation: a study of graph drawing aesthetics and algorithms. *Interacting with Computers*, 13(2):147–162, 2000.

[PW01] János Pach and Rephael Wenger. Embedding planar graphs at fixed vertex locations. *Graphs and Combinatorics*, 17(4):717–728, 2001.

[RCUG13] Eduardo Rivera-Campo and Virginia Urrutia-Galicia. A sufficient condition for the existence of plane spanning trees on geometric graphs. *Computational Geometry*, 46(1):1–6, 2013.

[RNG04] Md. Saidur Rahman, Takao Nishizeki, and Shubhashis Ghosh. Rectangular drawings of planar graphs. *J. Algorithms*, 50(1):62–78, 2004.

[RNN98] Md. Saidur Rahman, Shin-Ichi Nakano, and Takao Nishizeki. Rectangular grid drawings of plane graphs. *Comput. Geom.*, 10(3):203–220, 1998.

[SB92] Manojit Sarkar and Marc H. Brown. Graphical fisheye views of graphs. In *Proceedings of the Conference on Human Factors in Computing Systems CHI'92*, 1992.

[Sch90] Walter Schnyder. Embedding planar graphs on the grid. In *SODA*, pages 138–148. SIAM, 1990.

[Sch13] Marcus Schaefer. Toward a theory of planarity: Hanani-tutte and planarity variants. *J. of Graph Algorithms and Applications*, 17(4):367–440, 2013.

[Sch14] Marcus Schaefer. Picking planar edges; or, drawing a graph with a planar subgraph. In Christian A. Duncan and Antonios Symvonis, editors, *Graph Drawing - 22nd International Symposium, GD 2014, Würzburg, Germany, September 24-26, 2014, Revised Selected Papers*, pages 13–24, 2014.

[SG01] Vitaly Surazhsky and Craig Gotsman. Controllable morphing of compatible planar triangulations. *ACM Trans. Graph.*, 20(4):203–231, 2001.

[SG03] Vitaly Surazhsky and Craig Gotsman. Intrinsic morphing of compatible triangulations. *International Journal of Shape Modeling*, 9(2):191–202, 2003.

[Shn96] Ben Shneiderman. The eyes have it: A task by data type taxonomy for information visualizations. In *Proc. of the IEEE Symp. on Visual Lang.*, pages 336–343, 1996.

[SSS03] Marcus Schaefer, Eric Sedgwick, and Daniel Stefankovic. Recognizing string graphs in NP. *J. Comput. Syst. Sci.*, 67(2):365–380, 2003.

[Ste51] Sherman K. Stein. Convex maps. *Proceedings of the American Mathematical Society*, 2:464–466, 1951.

[Tar72] Robert Endre Tarjan. Depth-first search and linear graph algorithms. *SIAM J. Comput.*, 1(2):146–160, 1972.

[The13] The Khronos Group. WebGL, Web Graphic Library – OpenGL ES 2.0 for the Web, 2013. `http://www.khronos.org/webgl/` (acc. 2014).

[Tho83] Carsten Thomassen. Deformations of plane graphs. *J. Comb. Theory, Ser. B*, 34(3):244–257, 1983.

[Tho84] Carsten Thomassen. Plane representations of graphs. In J. A. Bondy and U. S. R. Murty, editors, *Progress in Graph Theory*, pages 43–69. Academic Press, New York, NY, 1984.

[Tut70] William T. Tutte. Toward a theory of crossing numbers. *Journal of Combinatorial Theory*, 8(1):45 – 53, 1970.

[TV85] Robert Endre Tarjan and Uzi Vishkin. An efficient parallel biconnectivity algorithm. *SIAM J. Comput.*, 14(4):862–874, 1985.

[Val81] Leslie G. Valiant. Universality considerations in VLSI circuits. *IEEE Transaction on Computers*, 30(2):135–140, 1981.

[Val98] Pavel Valtr. On geometric graphs with no k pairwise parallel edges. *Discrete & Computational Geometry*, 19(3):461–469, 1998.

[Wag36] Klaus Wagner. Bemerkungen zum vierfarbenproblem. *Jahresbericht der Deutschen Mathematiker-Vereinigung*, 2:26–32, 1936.

[Wag37] Klaus Wagner. Über eine eigenschaft der ebenen komplexe. *Mathematische Annalen*, 114(1):570–590, 1937.

[Whi33] Hassler Whitney. 2-isomorphic graphs. *Amer. J. Math.*, 55:245–254, 1933.

[Woe02] Gerhard J. Woeginger. Embeddings of planar graphs that minimize the number of long-face cycles. *Oper. Res. Lett.*, pages 167–168, 2002.

[WPCM02] Colin Ware, Helen C. Purchase, Linda Colpoys, and Matthew McGill. Cognitive measurements of graph aesthetics. *Information Visualization*, 1(2):103–110, 2002.

[WSP12] Andreas Wotzlaw, Ewald Speckenmeyer, and Stefan Porschen. Generalized k-ary tanglegrams on level graphs: A satisfiability-based approach and its evaluation. *Discrete Applied Mathematics*, 160(16-17):2349–2363, 2012.

[ZH03] Huaming Zhang and Xin He. Compact visibility representation and straight-line grid embedding of plane graphs. In *Algorithms and Data Structures, 8th International Workshop, WADS 2003, Ottawa, Ontario, Canada, July 30 - August 1, 2003, Proceedings*, volume 2748 of *LNCS*, pages 493–504. Springer, 2003.