ROMA
TRE
UNIVERSITÀ DEGLI STUDI

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

Small Screens and Large Graphs: Area-Efficient Drawings of Planar Combinatorial Structures

Fabrizio Frati

Small Screens and Large Graphs: Area-Efficient Drawings of Planar Combinatorial Structures

A thesis presented by
Fabrizio Frati
in partial fulfillment of the requirements for the degree of
Doctor of Philosophy
in Computer Science and Engineering

Roma Tre University
Dept. of Informatics and Automation

April 2009

Committee:
Prof. Giuseppe Di Battista

Reviewers:
Prof. Michael T. Goodrich
Prof. Ferran Hurtado

-Would you bring a message to your Lord?
-As you order. What should I say to Him?
-That the Black Warrior is arrived. Only this.

a Riccardo

Acknowledgments

My best acknowledgments go to my advisor Giuseppe Di Battista. I owe him what I know about graphs and algorithms, the insight that gave rise to our results, the possibility of traveling around (for conferences, workshops, visits, and vacations), the target of tackling harder and harder problems. But mostly I have to thank him for the interest and fun he puts at doing research; these are a clear answer to the question whether I will be bored of doing research in twenty years from now.

I would like to thank Michael Kaufmann, Stephen Kobourov, and János Pach that allowed me to collaborate with them during very pleasant and interesting visit periods that I spent at the University of Tübingen, at the University of Arizona, and at the New York University, respectively.

I would like to thank Michael Goodrich and Ferran Hurtado for carefully reviewing this thesis.

I would like to thank all my coauthors, I learned a lot from them and I enjoyed doing research so much also because it was with such amazing mates: Patrizio Angelini, Carla Binucci, Ulrik Brandes, Pier Francesco Cortese, Walter Didimo, Emilio Di Giacomo, Guido Drovandi, Cesim Erten, Alejandro Estrella-Balderrama, Joe Fowler, Markus Geyer, Luca Grilli, Carsten Gutwenger, Seok-Hee Hong, Katharina A. Lehmann, Giuseppe Liotta, Petra Mutzel, Maurizio Patrignani, Maurizio Pizzonia, Antonios Symvonis, Ioannis G. Tollis, and Francesco Trotta.

I would like to thank all the members of our research group creating that great atmosphere mixing friendship and collaboration that we have in our office: Luca Cittadini, Fabrizio Martorelli, Alessandro Marzioni, Bernardo Palazzi, Tiziana Refice, Massimo Rimondini, and Stefano Vissicchio.

Finally, I would like to thank all those I love. They are my life and hence they are fundamental to all I did, including this thesis, and to all I will do.

Contents

Introduction

Graphs are the most widely used data structures to represent relationships among objects. Maps, networks, circuits, molecules, compounds are a few examples of structures that are commonly represented by graphs. The clearest way to express the information conveyed in a graph is to visualize it. Namely, a *drawing* of a graph represents each object (in the graph terminology: *vertex*) of the graph as a point in the plane and each relationship (in the graph terminology: *edge*) between two objects as a line connecting the corresponding points. However, not every drawing can be regarded as a good representation of the graph. In fact, a drawing should be *readable*, that is, the human eye should be able to easily identify the relationships among the objects in the graph at the first glance to the drawing. Clearly, this is not a formal definition of what differentiates a good drawing from a bad drawing. However, a few topological and geometric features have been recognized and accepted as the criteria a drawing should satisfy in order to be readable.

Planarity is probably the best characteristic a drawing can have. The absence of intersections between the edges of the graph allows a viewer to easily distinguish the line representing any edge and hence to immediately understand which are the vertices connected by the edge. From a geometric perspective, it would be preferable that edges are drawn as *straight-lines*, namely edges bending and repeatedly changing direction are detrimental for the readability of the drawing. When the straight-line requirement can not be met, it would be still desiderable to have edges drawn as *poly-lines* bending only a limited number of times.

When the size of the graph to be represented is too large in order for the drawing to be constructed manually, there is a need for an algorithm automatically constructing such a drawing. *Graph Drawing* deals with the design of algorithms to automatically construct drawings of graphs. Usually a Graph Drawing algorithm takes as an input a graph, a set of requirements the drawing *must* satisfy (as being planar or having straight-line edges), and a set of aesthetics the drawing *should* satisfy as much as possible. The most important aesthetic a drawing should satisfy is probably the one of having a small area. In fact, automatic drawings usually have to be displayed on a computer screen of bounded size, hence they have to fit in the space available on the screen. The study of graph drawings in small area has been first motivated by the design of VLSI circuits and has attracted intense research efforts for almost thirty years now.

In this thesis we deal with algorithms and bounds for drawing graphs in small area. We mainly deal with planar graphs (Part I of the thesis), series-parallel graphs and outerplanar graphs (Part II), trees (Part III), and clustered graphs (Part IV), and for each of these graph classes we consider the problem of obtaining drawings in small area under a large number of drawing conventions (e.g., straight-line, poly-line, orthogonal, upward). We design several algorithms for the construction of graph drawings in small area and we obtain lower bounds for the area requirements of several drawing styles. The graph classes and the drawing conventions we consider are among the most commonly used for applications. Nevertheless, the beauty of some combinatorial, topological, and geometric problems concerning the construction of graph drawings in small area justifies their study even when looking at them from a purely theoretical point of view.

Part I of this thesis deals with drawings of planar graphs in small area.

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

In Chapter 2, we illustrate the state of the art on drawing *planar graphs* in small area under several drawing conventions, namely straight-line, poly-line, convex, and proximity drawings. Every planar graph admits a straight-line drawing in quadratic area and such a bound is worst-case asymptotically optimal, even for poly-line, orthogonal, and convex drawings. Determining the exact area bounds for straight-line drawings of plane graphs is a long-standing open problem, in which the gap between the best known upper bound and the best known lower bound (the latter one is a result of this thesis) is still quadratic in the number of vertices of the graph. Determining asymptotically optimal area bounds for strictly-convex drawings of triconnected planar graphs is another open problem that is attracting several research efforts.

In Chapter 3, we consider a proximity drawing standard that has recently attracted much attention, namely *greedy drawings* of graphs. We show that every triconnected planar graph admits a greedy drawing. The area requirements of the constructed drawings is exponential in the worst case. Determining the area requirements of greedy drawings of trees, triangulations, and triconnected planar graphs, determining whether convex greedy drawings exist for every triconnected planar graph, and characterizing the graphs admitting a greedy drawing are among the main open problems in this area.

Part II of this thesis deals with drawings of subclasses of planar graphs in small area.

In Chapter 4, we review the state of the art on drawing subclasses of planar graphs in small area. For several important classes of planar graphs, like bipartite planar graphs, four-connected planar graphs, and bounded-degree planar graphs, the area requirements of straight-line and poly-line drawings are still quadratic. Then, we consider *series-parallel graphs*, that are known to admit subquadratic-area poly-line drawings. We prove the first super-linear lower bound for the area requirements of poly-line drawings of series-parallel graphs. Such a lower bound is also the best known for straight-line drawings. Determining asymptotically-optimal area bounds for straight-line and poly-line drawings of series-parallel graphs are still intriguing open problems.

In Chapter 5, we show algorithms for drawing *outerplanar graphs*. In particular, it is known that every outerplanar graph can be drawn with poly-line edges in sub-quadratic area. We exhibit the first sub-quadratic area upper bound for straight-line drawings of outerplanar graphs. Algorithms achieving better area bounds are shown for balanced outerplanar graphs and for bounded-degree outerplanar graphs. Determining asymptotically-optimal area bounds for straight-line and poly-line drawings of outerplanar graphs seem still to be elusive tasks. In particular, no super-linear area lower bound and no algorithm achieving sub-quadratic area and constant aspect ratio are known at the state of the art.

Part III of this thesis deals with drawings of trees in small area.

In Chapter 6, we first review the state of the art on drawing *trees* in small area, taking under consideration straight-line, poly-line, orthogonal, and straight-line orthogonal drawings. Every bounded-degree tree admits straight-line drawings and orthogonal drawings in linear area. We present algorithms for constructing straight-line orthogonal drawings of binary and ternary trees. In particular, we show how to construct straight-line orthogonal order-preserving drawings of binary trees and straight-line orthogonal drawings of ternary trees in sub-quadratic area. Whether every tree admits a linear-area straight-line drawing, and whether every binary tree admits a linear-area straight-line orthogonal drawing are among the main problems in this area.

In Chapter 7, we consider the problem of constructing small-area upward drawings of *directed trees.* Unlike non-directed planar graphs, upward planar directed graphs may require exponential area in any upward straight-line drawing. On the other hand, quadratic area is a tight bound for constructing upward poly-line drawings of upward planar directed graphs. We show that directed trees can be drawn in optimal sub-quadratic area while exponential area is sometimes necessary if an order of the neighbors of each node is fixed in advance. We also extend such a lower bound to upward planar directed bipartite graphs and upward planar directed outerplanar graphs. For several classes of upward planar directed graphs, like directed balanced trees, determining the area requirements for upward planar drawings is still an open problem.

In Chapter 8, we consider the problem of constructing drawings of *minimum spanning trees*. It is known that no tree having a vertex of degree at least seven can be realized as a minimum spanning tree, that deciding whether degree-six trees can be realized as minimum spanning trees is a difficult problem, and that every tree of maximum degree five can be realized as a minimum spanning tree. However, all known algorithms for laying out degree-five minimum spanning trees require exponential area. We show that degree-tree and degree-four trees have polynomial-area drawings as minimum spanning trees. It is an obvious and important open problem to determine whether drawings of degree-five minimum spanning trees can be constructed in polynomial-area.

Part IV of this thesis deals with drawings of clustered graphs.

In Chapter 9, we consider the problem of constructing small-area drawings of *clustered trees* under several drawing standards. It is known that every clustered graph admits a straight-line c-planar drawing in which the clusters are represented by convex polygons. However, the area requirement of such drawings is exponential in the worst case. We show that c-connected clustered trees admit quadratic-area straight-line c-planar drawings in which clusters are represented by rectangles. However, we prove that exponential area is required for c-planar straight-line drawings of non-c-connected clustered trees in which clusters are drawn as convex polygons. Several problems concerning the area requirements of clustered trees remain open. Among them, whether every c-connected tree admits a straight-line order-preserving drawing in polynomial area seems to be particularly appealing.

In Chapter 10, we consider the problem of deciding whether a clustered graph admits a *c-planar drawing*. Such a problem is one of the most studied in the recent Graph Drawing literature. Although the complexity of such a problem remains unknown in the general case, we provide a characterization and a linear-time testing algorithm for deciding the c-planarity of embedded flat clustered graphs in which all faces have at most five incident vertices. Such results are valid, more in general, if each candidate saturating edge, i.e., each edge that can be added to the graph to make it c-connected, crosses at most one distinct candidate saturating edge. Determining the time complexity of testing the c-planarity of a clustered graph remains an fascinating open problem.

Part I

Planar Graphs

Chapter 1

Graph Preliminaries and Definitions

In this chapter, we introduce some preliminaries and definitions about graphs and their drawings. 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., [Har72, BM76, NC88, Die05]), to books on Algorithms (e.g., [Eve79, AHU83, CLRS01, GT02]), and to books on Computational Geometry (e.g., [PS85, Ede87, dvKOS00]). The book of Di Battista, Eades, Tamassia, and Tollis [DETT99] is usually considered as *the book* on Graph Drawing. Other excellent books that specifically deal with Graph Drawing are [KW01, NR04].

1.1 Introduction

A *graph* is a pair (V, E), where V is a set of *vertices* or *nodes*, and E is a multiset of *unordered* pairs of vertices, called *edges* or *arcs*. A *directed graph* is a pair (V, E), where V is a set of *vertices* or *nodes*, and E is a multiset of *ordered* pairs of vertices, called *edges* or *arcs*. The graph obtained from a digraph G by considering its edges without orientation is called the *underlying graph* of G. In the following, unless otherwise specified, when we say "graph" we refer to an undirected graph. The vertices u and v composing a pair $e = (u, v) \in E$ are *incident* to e, and edge e is *incident* to u and v. Two vertices are *adjacent* when they are incident to the same edge, and two edges are *adjacent* when they are incident to the same vertex. The *end-vertices* of an edge (u, v) are vertices u and v, which are also said to be *neighbors*.

A *self-loop* in a graph (V, E) is an edge $(u, u) \in E$. A set of *multiple edges* or *parallel edges* in a graph (V, E) is a set of edges connecting the same two vertices $u, v \in V$. A graph is *simple* when it contains no self-loops and no multiple edges. In the following, unless otherwise specified, we always refer to simple graphs.

The *degree of a vertex* is the number of edges incident to the vertex. The *degree of a graph* is the maximum among the degrees of its vertices.

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. A *path* is a tree such that each vertex has degree at most two.

A graph $G'(V', E')$ is a *subgraph* of a graph $G(V, E)$ if $V' \subseteq V$ and $E' \subseteq E$. A subgraph $G'(V', E')$ of a graph $G(V, E)$ is *induced* by V' if, for every edge $(u, v) \in E$ such that $u, v \in V'$, $(u, v) \in E'$. A graph $G'(V', E')$ is a *spanning subgraph* of $G(V, E)$ if it is a subgraph of G and $V' = V$.

A *subdivision* of a graph G is a graph G' that can be obtained by replacing each edge of G by a path of arbitrary length. The *contraction* of two vertices u and v is the replacement of u and v by a single vertex w, that is adjacent to every vertex u and v are adjacent to. A *minor* of a graph G is any graph that can be obtained from G by a sequence of the following three operations:

- Removing a vertex.
- Removing an edge.
- Contracting an edge.

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 open Jordan curve between its *endpoints*, i.e., the points to which the end-vertices of the edge have been mapped. It is important to notice the difference between a graph, that is an abstract structure corresponding to a relationship among objects, and its drawing, that is a graphical representation of the graph.

The rest of the chapter is organized as follows. In Sect. 1.2 we introduce preliminaries and definitions about planar graphs and planar embeddings; in Sect. 1.3 we introduce preliminaries and definitions about the maximality and the connectivity of (planar) graphs; in Sect. 1.4 we introduce preliminaries and definitions about subclasses of planar graphs; finally, in Sect. 1.5 we introduce preliminaries and definitions about Graph Drawing.

1.2 Planar Drawings, Embeddings, and Graphs

In this section we present some preliminaries and definitions about planar graphs, planar embeddings, and planar drawings.

A drawing is *planar* when no two edges intersect except, possibly, at common endpoints. A *planar graph* is a graph admitting a planar drawing. 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 fact, a planar drawing of a graph provides extremely high readability of the combinatorial structure of the graph [PCJ97, Pur00]. See Fig. 1.1 for a comparison between a non-planar and a planar drawing.

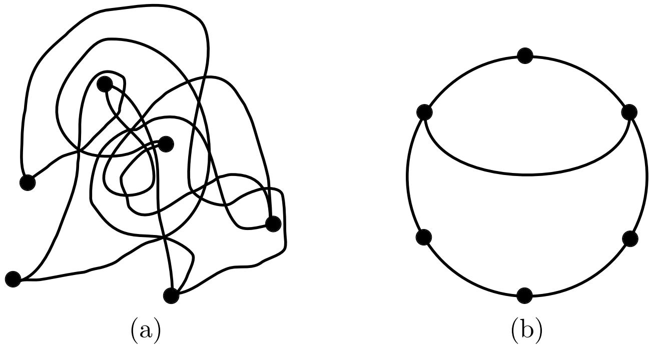


Figure 1.1: (a) A non-planar drawing of a graph. (b) A planar drawing of the same graph.

A planar drawing of a graph determines a circular ordering of the edges incident to each vertex. Two drawings of the same graph are *equivalent* if they determine the same circular ordering around each vertex. A *planar 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 if the vertex (resp. the edge) belongs to the cycle delimiting the face. Two faces are *adjacent* when they share an edge. The unbounded face is the *outer face* (sometimes also called the *external face*), while the bounded faces are the *internal faces*. The outer face of a graph G is denoted by $f(G)$. 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^* of an embedded planar graph G has a vertex for each face of G and has an edge (f, g) for each two faces f and g of G sharing an edge. Fig. 1.2 shows an embedded planar graph 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.

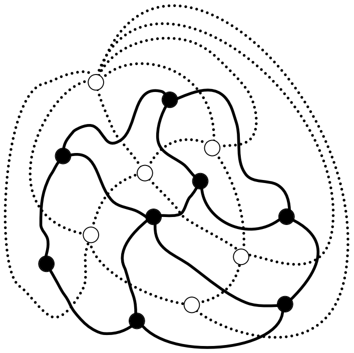


Figure 1.2: An embedded planar graph, whose vertices are represented by black circles and whose edges are represented by thick lines, and its dual graph, whose vertices are represented by white circles and whose edges are represented by dotted lines.

Planar graphs have been nicely characterized by Kuratowski [Kur30] as the graphs containing no subdivision of K_5 and no subdivision of $K_{3,3}$, and by Wagner as the graphs containing 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 in [HT74]. Linear-time algorithms for testing the planarity of a graph are also presented, e.g., in [BL76, ET76, dR82]. The cited planarity testing algorithms can be suitably modified in order to compute planar embeddings in the case the graph is found to be planar. If an embedding of a graph is fixed, then linear time still suffices to test if the embedding is planar [Kir88].

1.3 Maximality and Connectivity

In this section we present some preliminaries and definitions about maximal planar graphs, about the connectivity of graphs, and about the data structures to handle the connectivity properties of a planar graph.

A plane graph is *maximal* (or equivalently is a *triangulation*) when all its faces are delimited by cycles of three vertices. Fig. 1.3 (a) shows a maximal plane graph. A planar graph is *maximal* when it can be embedded as a triangulation. An algorithm for drawing planar graphs can usually assume to deal with maximal planar graphs. In fact, any planar graph can be augmented to a maximal planar graph by adding some *dummy edges* to the graph. Then the algorithm can draw the maximal planar graph, and finally the inserted dummy edges can be removed obtaining a drawing of the input graph. A plane graph is *internally-triangulated* when all its internal faces are cycles of three vertices. Fig. 1.3 (b) shows an internally-triangulated plane graph. A *chord of a cycle* is an edge connecting two non-consecutive vertices of the cycle. A *chord of a plane graph* is a chord of the cycle delimiting its outer face.

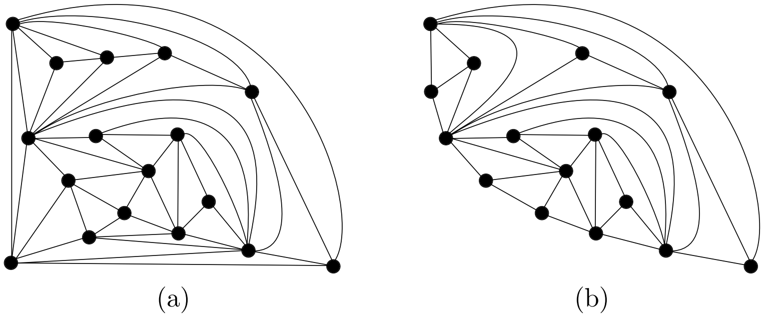


Figure 1.3: (a) A drawing of a maximal plane graph. (b) A drawing of an internally-triangulated plane graph.

A graph is *connected* if every pair of vertices of G is connected by a path. A k*-connected* graph G is such that removing any $k-1$ vertices leaves G connected; 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 and separating 2-sets are also called *cutvertices* and *separation pairs*, respectively. Hence, a connected graph is biconnected if it has no cutvertices, and it is triconnected if it has no separation pairs. The maximal biconnected subgraphs of a graph are its *blocks*. Each edge of G falls into a single block of G, while cutvertices are shared by different blocks.

Data structures exist to efficiently handle the decomposition of a connected graph into biconnected components and the subdivision of a biconnected graph into triconnected components. Such data structures are called *BC-trees* and *SPQR-trees*, respectively. They were introduced by Harary and Prins in [HP66], and by Di Battista and Tamassia in [DT90, DT96b, DT96a], respectively. In the following, we present the definition of BC-trees, that will be used throughout this thesis, while we skip the definition of SPQR-trees, which is more involved and not strictly related to the contents of this thesis.

The *block-cutvertex tree*, or BC-tree, of a connected graph G is a tree with a B-node for each block of G and a C-node for each cutvertex of G. Edges in the BC-tree connect each B-node μ to the C-nodes associated with the cutvertices in the block of μ.

A graph G is *edge k-connected* if the removal of any $k-1$ edges leaves G connected. A *separating edge* (sometimes also called *bridge*) is an edge whose removal disconnects G.

1.4 Classes of Planar Graphs

In this section we present some preliminaries and definitions about sub-classes of planar graphs.

A *tree* is a connected acyclic graph. A *leaf* in a tree is a node of degree one. A *caterpillar* C is a tree such that the removal from C of all the leaves and of their incident edges turns C into a path (see Fig. 1.4 (a)). A *spider tree* is a tree having only one vertex of degree greater than two (see Fig. 1.4 (b)).

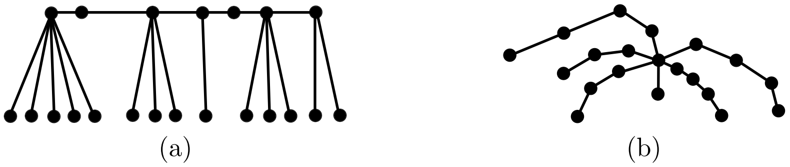


Figure 1.4: (a) A caterpillar. (b) A spider tree.

A *rooted tree* is a tree with one distinguished node called *root*. In a rooted tree T the root is denoted by $r(T)$. In a rooted tree each node v at distance (i.e., length of the shortest path) d from the root is the child of the only node at distance $d-1$ from the root v is connected to. A *binary tree* (a *ternary tree*) is a rooted tree such that each node has at most two children (resp. three children). Binary and ternary trees can be supposed to be rooted at any node of degree at most two and three, respectively. Every tree of degree at most three (at most four) can be rooted to a binary tree (resp. ternary tree). A *spine* in T is a path connecting $r(T)$ to a leaf. A *double-spine* in T is a path connecting two leaves and passing through $r(T)$. The *height* of a rooted tree is the maximum number of nodes in any spine.

A *complete binary tree* (a *complete ternary tree*) is a rooted binary tree such that each non-leaf node has exactly two children (resp. three children) and such that each spine has the same number of nodes. Note that the height of an n-node complete binary tree is $\log_2(n+1)$, while the one of a complete ternary tree is $\log_3(2n+1)$. A class of trees is *balanced* if there exists a constant $k > 0$ such that the height of each n-node tree of the class is bounded by $k \log n$. In what follows when we say *balanced tree* we refer to a tree of a balanced class. *AVL trees* and *Fibonacci trees* are balanced families of trees.

A tree is *ordered* if an order of the children of each node (i.e., a planar embedding) is specified. For an ordered binary tree we talk about *left* and *right child*. For an ordered ternary tree we talk about *left*, *middle*, and *right child*. Removing a non-leaf node u from a tree disconnects the tree into connected components. The ones containing children of u are *subtrees* of u. If the tree is ordered and binary (ternary), the subtrees rooted at the left and right child (resp. at the left, middle, and right child) of a node u are the *left* and the *right subtree* of u (resp. the *left*, the *middle*, and the *right subtree* of u), respectively. Removing a path $\mathcal{P}$ from a tree disconnects the tree into connected components. The ones containing children of nodes in $\mathcal{P}$ are subtrees of $\mathcal{P}$. If the tree is ordered and binary (resp. ternary), then each component is a left or right subtree (resp. a left, middle, or right subtree) of $\mathcal{P}$, depending on whether the root of such subtree is a left or right child of a node in $\mathcal{P}$ (resp. is a left, middle, or right child), respectively. We denote by $|T|$ the number of nodes in a tree T. The subtree of a rooted tree T rooted at a vertex v is denoted by $T(v)$. The *heaviest tree* in a set of trees is the one with the greatest number of nodes.

Two binary trees are *simply isomorphic* if they are both empty or have isomorphic left subtrees and isomorphic right subtrees.

An *outerplanar graph* is a graph that contains no K_4-minor and no $K_{2,3}$-minor. However, outerplanar graphs due their name to the fact that they have the following embedding properties. An *outerplane graph* (see Fig. 1.5 (a)) 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. A *maximal outerplane graph* (see Fig. 1.5 (b)) 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.

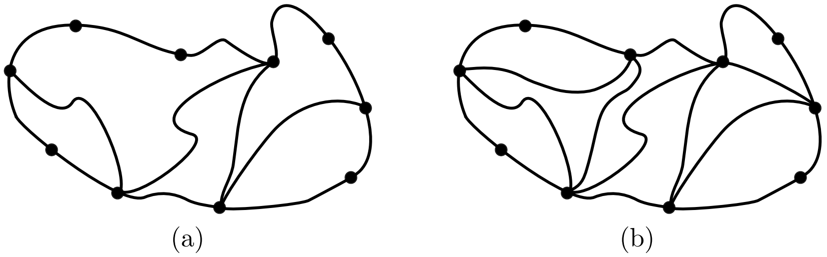


Figure 1.5: (a) A drawing of an outerplane graph. (b) A drawing of a maximal outerplane graph.

The *dual graph* of an outerplane graph G is a tree, when not considering the vertex corresponding to the outer face of G. Hence, when dealing with outerplanar graphs, we talk about the *dual tree* of an outerplanar graph (meaning the dual graph of an outerplane embedding of the outerplanar graph). The nodes of the dual tree of a maximal outerplane graph G have degree at most three. Hence the dual tree of G can be rooted to be a binary tree. In fact, when dealing with outerplanar graphs, it is sometimes convenient to assume that their dual trees are rooted. For this purpose, consider any maximal outerplanar graph G. We can select an edge (u, v) incident to the outer face of (an outerplane embedding of) G and root the dual binary tree T of G at the internal face f_0 containing (u, v) (see Fig. 1.6). Let w be the third vertex of f_0. We call u and v *poles* and w *central vertex* of G. We also call u *left vertex* and v *right vertex* of G. Consider a face f of G and suppose that f is composed by edges (v_1, v_2), (v_2, v_3), and (v_3, v_1), in this clockwise order around f. Also, suppose that the parent of f in T and f share edge (v_1, v_2) or that (f is the root) $(v_1, v_2) = (u, v)$. The face sharing with f (if any) edge (v_3, v_1) is the *left child* of f, while the face sharing with f (if any) edge (v_2, v_3) is the *right child* of f.

Two maximal outerplanar graphs are *isomorphic* if their dual trees are isomorphic.

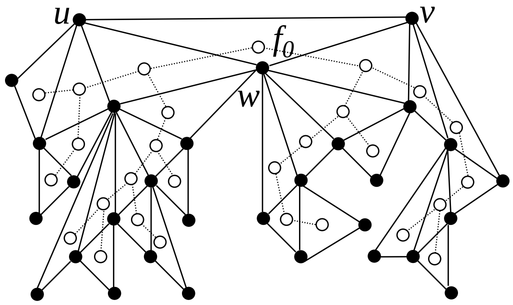
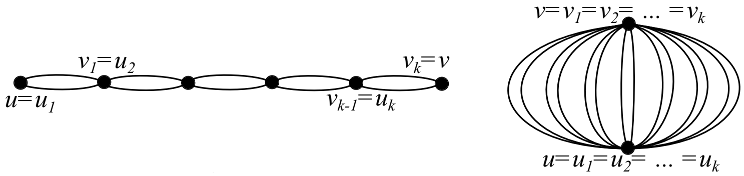


Figure 1.6: A drawing of an outerplanar graph and of its dual tree.

A *balanced outerplanar graph* is an outerplanar graph whose dual tree can be rooted to a balanced binary tree. The *height* of a balanced outerplanar graph is the height of its dual tree. A *complete outerplanar graph* is a balanced outerplanar graph whose dual tree can be rooted to a complete binary tree.

A *series-parallel graph* is a graph that contains no K_4-minor. However, series-parallel graphs due their name to the fact that they can be inductively constructed as follows. An edge (u, v) is a series-parallel graph with *terminals* u and v. Denote by u_i and v_i the terminals of a series-parallel graph G_i. Then, a *series composition* (see Fig. 1.7 (a)) of a sequence $G_1, G_2, \ldots, G_k$ of series-parallel graphs, with $k \geq 2$, constructs a series-parallel graph that has terminals $u = u_1$ and $v = v_k$, that contains graphs G_i as subgraphs, and such that vertices v_i and u_{i+1} have been identified to be the same vertex, for each $i = 1, 2, \ldots, k-1$. A *parallel composition* (see Fig. 1.7 (b)) of a set $G_1, G_2, \ldots, G_k$ of series-parallel graphs, with $k \geq 2$, constructs a series-parallel graph that has terminals $u = u_1 = u_2 = \cdots = u_k$ and $v = v_1 = v_2 = \cdots = v_k$, that contains graphs G_i as subgraphs, and such that vertices $u_1, u_2, \cdots, u_k$ (vertices $v_1, v_2, \cdots, v_k$) have been identified to be the same vertex. A *maximal series-parallel graph* is such that all its series compositions construct a graph out of exactly two smaller series-parallel graphs G_1 and G_2, and such that all its parallel compositions have a component which is the edge between the two terminals. Every series-parallel graph can be augmented to maximal by adding dummy edges to it.


$v_1 = u_2$
$v_k = v$
$u = u_1$
$v_{k-1} = u_k$
$v = v_1 = v_2 = \ldots = v_k$
$u = u_1 = u_2 = \ldots = u_k$

(a) (b)

Figure 1.7: (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.

A *bipartite graph* G is such that its vertex set V can be partitioned into two subsets V_1 and V_2 so that every edge of G is incident to a vertex of V_1 and to a vertex of V_2. A *bipartite planar graph* is both bipartite and planar. A *maximal bipartite planar graph* admits a planar embedding in which all its faces have exactly four incident vertices. Every bipartite planar graph can be augmented to maximal by adding dummy edges to it.

A *Hamiltonian cycle (path)* in a graph G is a simple cycle (resp. path) passing through all the vertices of G. A *Hamiltonian graph* is a graph containing a Hamiltonian cycle. Every 4-connected planar graph is Hamiltonian, while the converse is not true. Further, every biconnected outerplanar graph G has exactly one Hamiltonian cycle, namely the one delimiting the face to which all the vertices of G are incident in a outerplanar embedding of G.

1.5 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*.

Drawing Conventions. There are several widely-studied drawing conventions in the Graph Drawing literature. In the following we show a list of the ones that are of interest for this thesis:

- *Poly-line Drawings:* Each edge is represented by a sequence of consecutive segments.
- *Straight-line Drawings:* Each edge is represented by a segment. A straight-line drawing is shown in Fig. 1.8 (a).
- *Grid Drawings:* Vertices and bends have integer coordinates.
- *Orthogonal Drawings:* Each edge is represented by a sequence of horizontal and vertical segments. An orthogonal drawing is shown in Fig. 1.8 (b).

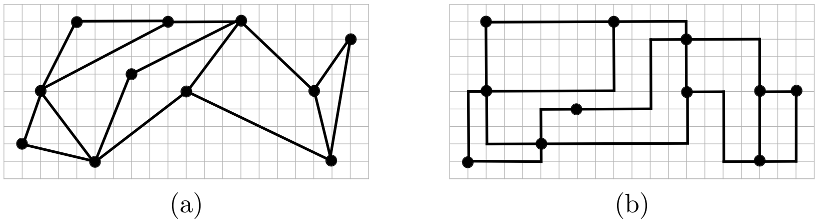


Figure 1.8: Two planar grid drawings of the same graph. (a) A straight-line drawing. (b) An orthogonal drawing.

- *Planar Drawings:* No two edges intersect.
- *Convex Drawings:* Planar straight-line drawings in which every face is represented by a convex polygon.
- *Upward Drawings:* For directed graphs, each directed edge is represented by a monotonically-increasing curve. For rooted (undirected) trees, the

definition changes slightly. In fact, an *upward drawing* of a rooted tree is such that each edge from a node to its child is represented by a monotonically-non-decreasing curve, while a *strictly-upward drawing* of a rooted tree is such that each edge from a node to its child is represented by a monotonically-increasing curve.

- *Order-Preserving Drawings:* The order of the edges incident to each vertex is fixed in advance.

- *Visibility Representations:* Each vertex u is represented by a horizontal segment $\sigma(u)$, and each edge (u, v) is represented by a vertical segment connecting a point of $\sigma(u)$ with a point of $\sigma(v)$.

- *Proximity Drawings:* Given a definition of *proximity* (for example, two points p_1 and p_2 can be defined to be *proximate* if no point lies in the circumference having p_1 and p_2 as antipodal points), the *proximity graph* of a set of points is the graph with a vertex for each point of the set, and 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 Γ of G such that the proximity graph of the set of points on which the vertices of G are drawn in Γ coincides with G itself.

Straight-line and orthogonal drawings are special cases of poly-line drawings. Straight-line drawings (which constitute a great part of the topics covered by this thesis) are probably the most studied type of drawings, since they are very natural and they provide high readability; references to such drawings can be found in several graph theory books and papers.

Drawing Aesthetics. There are several widely-studied drawing aesthetics in the Graph Drawing literature. In the following we show a list of the ones that are of interest for topics related to this thesis:

- *Area:* Minimization of the area of the drawing. The *bounding box* $B(\Gamma)$ of a drawing Γ is the smallest rectangle with sides parallel to the axes that covers Γ completely. We denote with $b(\Gamma)$, $t(\Gamma)$, $l(\Gamma)$ and $r(\Gamma)$ the bottom, top, left and right side of $B(\Gamma)$, respectively. The *height* (*width*) of Γ is the height (resp. width) of $B(\Gamma)$. The *area* of a drawing is the area of $B(\Gamma)$. Designing algorithms for constructing small area drawings is usually regarded as a very important task. In fact, the screens on which the graphs have to be displayed are bounded, hence the more an algorithm constructs compact drawings, the more large are the graphs that can be shown on a screen. Most of this thesis deals with the construction of drawings in small area. Notice that the concept of area of a drawing only makes sense once fixed a *resolution rule*, i.e., a rule that does not allow vertices to be arbitrarily close and edges to be arbitrarily short. In fact, without any of such rules, one could just construct arbitrarily small drawings and enclose them in an arbitrarily small area. A good model to ensure that the drawings satisfy such rules is the one of constructing grid drawings in which, by definition, every two vertices have distance at least one unit.

- *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

Straight-line, Poly-line, Convex, and Proximity Drawings of Planar Graphs

In this chapter, we illustrate the state of the art on drawing planar graphs in small area under several drawing conventions. We mainly focus on straight-line drawings, poly-line drawings, convex drawings, and proximity drawings. We give a sketch of the algorithms achieving the best known upper bounds at the state of the art, and we show the graphs that give rise to the best lower bounds at the state of the art.

2.1 Introduction

Constructing planar straight-line drawings of planar graphs is one of the most studied topics in Graph Drawing. In fact, straight-line planar drawings are aesthetically pleasant and easily readable by the human eye, hence they are among the best drawing styles to be used for computer applications; moreover, understanding the geometric properties of planar graphs is of course an interesting mathematical issue, even when not motivated by any practical reason.

A ground-breaking result of the last century is that *every plane graph admits a planar straight-line drawing*. Such a result was independently proved by Wagner [Wag36], by Fary [Far48], and by Stein [Ste51]. However, that every plane graph admits a planar straight-line drawing is often regarded as the "Fary's theorem" (and a planar straight-line drawing is sometimes regarded as a "Fary embedding"), probably because of the beauty and the simplicity of the Fary's proof.

Although the algorithm of Fary is very easy to understand and to implement, it turns out to be useless for the visualization of planar graphs with a large number of vertices. In fact, it has the great drawback of assigning *real coordinates* to the vertices. Such an assignment implies that two vertices can be arbitrarily close or arbitrarily far in the drawing, i.e., the ratio between the longest and the smallest edge in the drawing can be exponential in the number of nodes of the graph, which is surely detrimental for the readability of the drawing. Moreover, a computer storing the coordinates assigned to the vertices would need a polynomial number of bits, while coordinates needing a logarithmic number of bits to be represented on a computer would be desiderable.

Hence, the question was raised by Rosenstiehl and Tarjan in [RT86] whether planar straight-line drawings of n-vertex planar graphs can be realized on a *grid* of side length bounded by n^k, for some fixed k.

The problem was first solved in a paper by de Fraysseix, Pach, and Pollack appeared at the *Symposium on Theory of Computing* in 1988 [dPP88] (and further published in [dPP90]). The authors showed that every n-vertex plane graph admits a planar straight-line drawing on a $(2n-4) \times (n-2)$ grid. Such a result uses as main tool a very natural and simple order between the vertices of a maximal plane graph, namely the *canonical ordering.* Such an ordering is used to incrementally construct a straight-line drawing, by adding the vertices one at a time to the partially constructed drawing.

Only two years later, Schnyder presented, at the first *Symposium on Discrete Algorithms* [Sch90], an algorithm improving the grid size down to $(n-2) \times (n-2)$. The result relies on techniques completely different from the ones of de Fraysseix *et al.* In fact, Schnyder proved that the internal edges of a maximal plane graph G can be partitioned into three trees representing three partial orders between the vertices of G. Such orders can be used to assign three-dimensional integer coordinates to the vertices of G, thus obtaining a straight-line planar drawing of G lying on a plane in the three-dimensional space. Finally, such a drawing can be projected on the xy-plane, thus obtaining a straight-line planar drawing of G on a $(n-2) \times (n-2)$ grid.

An area of $\lfloor 2(n-1)/3 \rfloor \times (4\lfloor 2(n-1)/3 \rfloor - 1)$ was shown to be sufficient for constructing straight-line drawings of plane graphs by Chrobak and Nakano [CN98]. The width of the drawing is the minimum possible. Bounds of $(n-2-\Delta^-) \times (n-2-\Delta^-)$ and $(n-2-\Delta) \times (n-2-\Delta)$ were achieved by Zhang and He in [ZH03], and by Bonichon, Felsner, and Mosbah in [BFM07].

Δ^- and Δ are distinct parameters of the plane graph, that however can be equal to 0 in the worst case.

A grid of quadratic size is asymptotically the best possible, since there exist plane graphs, and even planar graphs, requiring such an area in any planar grid drawing. this was first observed by Valiant in [Val81]. The lower bound was then improved to $\left(\frac{2n}{3}-1\right) \times \left(\frac{2n}{3}-1\right) = \frac{4n^2}{9} - \frac{4n}{3} + 1$ by de Fraysseix *et al.* [dPP90]. Currently, the best lower bound is, as far as we know, $\frac{4n^2}{9} - \frac{2n}{3}$, that has been proved by Frati and Patrignani in [FP07].

While the determination of the exact bounds for the area requirements of straight-line drawings of planar graphs still requires research efforts, tight bounds on the area requirements of *poly-line drawings* of plane graphs are known, up to the constants. In fact, Bonichon, Le Saëc, and Mosbah proved in [BSM02] that every planar graph has a poly-line drawing in $\frac{4(n-1)^2}{9}$ area, which matches the lower bound of de Fraysseix, Pach, and Pollack [dPP90].

Concerning *convex drawings*, quadratic area is sufficient for all triconnected plane graphs, if angles equal to π radiants are allowed [Kan96, CK97, ST92, DTL99, BFM07]. Clearly, such a bound is asymptotically tight. Otherwise, i.e., if all the angles of the internal faces of the drawing have to be strictly less than π, then all the n-vertex triconnected plane graphs can be drawn in $O(n^4)$ area, as shown by Barany and Rote in [BR06], while cubic area is sometimes necessary [And63, BP92, BT04].

Concerning *proximity drawings*, there exist meaningful definitions of proximity for which exponential area is sometimes required for drawing planar graphs (as proved by Liotta *et al.* in [LTTV97] for Gabriel drawings). On the other hand, some polynomial upper bounds are known for drawing trees under several definitions of proximity [PV04, FK08].

The rest of the chapter is organized as follows. In Sect. 2.2, we expose the state of the art on straight-line drawings of planar graphs in small area, by briefly discussing the algorithms of Fary [Far48], of de Fraysseix, Pach, and Pollack [dPP90], and of Schnyder [Sch90], and by showing some planar graphs providing the best known area lower bounds; in Sect. 2.3, we expose the state of the art on poly-line drawings of planar graphs in small area; in Sect. 2.4, we expose the state of the art on convex drawings of planar graphs in small area; in Sect. 2.5, we expose the state of the art on proximity drawings of planar graphs in small area. Finally, in Sect. 2.6 we conclude and present some open problems.

2.2 Straight-line Drawings

In this section, we sketch the most-widely known algorithms for constructing straight-line drawings of planar graphs. In Sect. 2.2 we illustrate Fary's algorithm, in Sect. 2.2 we show de Fraysseix, Pach and Pollack's algorithm, and in Sect. 2.2 we present Schnyder's algorithm. Finally, in Sect. 2.2, we discuss lower bounds for the area requirements of straight-line drawings of plane and planar graphs. More results on straight-line drawings of planar graphs (and their subclasses) in small area are exhibited in Chapters 4, 5, 6, 7, 8, and 9.

Fary's Algorithm

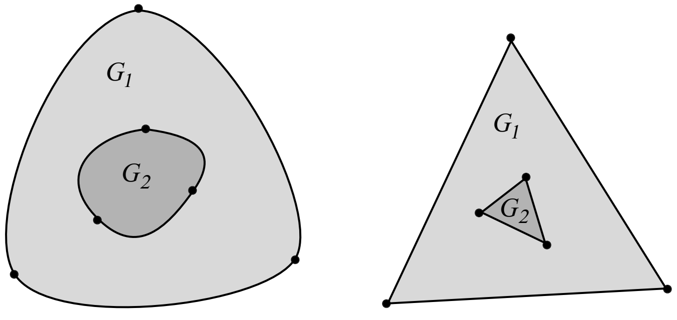
In this section we illustrate Fary's algorithm [Far48]. Such an algorithm proves that a straight-line drawing of a plane graph G can be constructed for an arbitrary drawing of $f(G)$. This is stated more precisely in the following:

Theorem 2.1 *(Fary [Far48]) Let G be any maximal plane graph and let Δ be any triangle in the plane. Then G admits a straight-line drawing Γ with external face Δ.*

The proof of the theorem is by induction on the number of nodes of G. In the base case, G has three vertices and the proof is trivial. Suppose that G has $n > 3$ vertices. Distinguish two cases:

- If G contains a separating 3-cycle c, then consider graphs G_1, obtained from G by removing all the vertices internal to c and consider graph G_2, obtained from G by removing all the vertices external to c (see Fig. 2.1 (a)). By the definition of separating 3-cycle, both graphs have less than n vertices. Apply the inductive hypothesis to construct a straight-line planar drawing Γ_1 of G_1 in which $f(G_1)$ is drawn as Δ. Since c is the cycle delimiting the border of an internal face of G_1, c is represented by a triangle Δ' in Γ_1. Apply again the inductive hypothesis to construct a straight-line planar drawing Γ_2 of G_2 in which $f(G_2)$ is drawn as Δ'. Plug Γ_2 inside Γ_1 by gluing the two drawings along the common face delimited by c and represented by Δ' in both drawings (see Fig. 2.1 (b)).

- If G does not contain any separating 3-cycle, i.e. G is 4-connected, consider any internal vertex u of G and consider any neighbor v of u. Construct an $(n-1)-$vertex graph G' by removing u and all its incident edges



(a) (b)

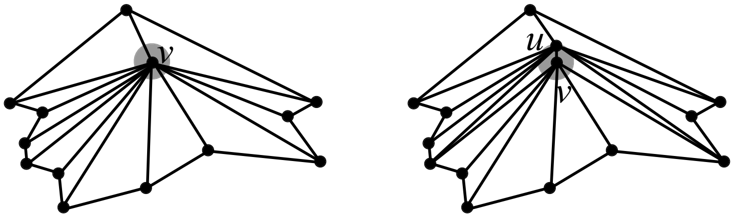
Figure 2.1: (a) A plane graph G containing a separating 3-cycle c. (b) Inductive construction of a straight-line drawing of G.

from G, and by inserting dummy edges between v and all the neighbors of u in G, except for the two vertices v_1 and v_2 forming faces with u and v. The resulting graph is clearly planar. Further, it is simple. In fact, if G' is not simple, then G contains an edge between v and one of the neighbors of u, say w, distinct from v_1 and v_2. However, this would imply that (u, v, w) is a separating 3-cycle, contradicting the assumptions. Hence, the inductive hypothesis can be applied to construct a straight-line planar drawing Γ' of G' in which $f(G')$ is drawn as Δ (see Fig. 2.2 (a)). Further, dummy edges can be removed and vertex u can be introduced in Γ' together with its incident edges, without altering the planarity of Γ'. In fact, u can be placed at a suitable point in the interior of a small disk centered at v, thus obtaining a straight-line drawing Γ of G in which $f(G)$ is represented by Δ (see Fig. 2.2 (b)).

de Fraysseix, Pach, and Pollack's Algorithm

In this section we describe the algorithm presented by de Fraysseix, Pach, and Pollack in [dPP90]. The algorithm relies on two main ideas.

First, it is shown that any maximal plane graph can be constructed by adding vertices one at a time, in an order such that a new vertex is always added in the external face of the currently constructed plane graph, and such that the current graph is always biconnected. Such an ordering of the vertices of a plane graph has been called *canonical ordering*. Canonical ordering and



(a) (b)

Figure 2.2: (a) A straight-line drawing of the plane graph G' obtained from G by *contracting* vertices u and v into a single vertex v. (b) Inductive construction of a straight-line drawing of G.

variations of the canonical ordering have been studied also in [Kan96].

Second, it is shown how to construct a straight-line drawing of a plane graph by adding vertices to a partially constructed drawing so that the order in which the vertices are introduced in the drawing is a canonical ordering of the graph, and so that the drawing always maintains some strong geometric invariants, among which the fact that the border of the outer face of the partially constructed drawing is represented by segments having slopes equal to either $\pi/4$ radiants or $-\pi/4$ radiants. In the following, the two results are stated more precisely.

Canonical Ordering of a Plane Graph. Let G be a maximal plane graph with outer face (u, v, w). Then the following holds (refer to Fig. 2.3):

Lemma 2.1 *(de Fraysseix, Pach, and Pollack [dPP90]) There exists a labelling of the vertices* $v_1 = u, v_2 = v, v_3, \ldots, v_n = w$ *meeting the following requirements for* $4 \leq k \leq n$*:*

- *(i) The subgraph* $G_{k-1} \subset G$ *induced by* $v_1, v_2, \cdots, v_{k-1}$ *is biconnected, and the boundary of its outer face is a cycle* C_{k-1} *containing edge* (u, v)*.*
- *(ii) Vertex* v_k *is in the outer face of* G_{k-1}*, and its neighbors in* G_{k-1} *form an (at least 2-element) subinterval of the path* $C_{k-1} - (u, v)$*.*

The proof of the previous lemma mainly relies on an auxiliary lemma stating that, given a simple plane graph G and given a simple cycle c of G, there exists a vertex of c not incident to any chord inside c. This allows de Fraysseix *et al.* to prove the existence of a canonical ordering for a given maximal plane graph

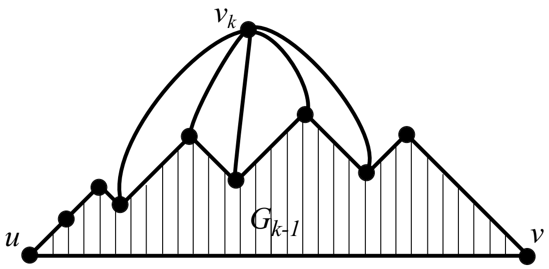


Figure 2.3: An illustration of the canonical ordering of a maximal plane graph.

G by reverse induction, namely by setting $v_n = w$, by assuming that G_{k-1} has been already determined (which is true in the base case $k = n$) with outer face delimited by a cycle C_{k-1}, and by then setting v_{k-1} equal to a vertex of G_{k-1} that is not incident to a chord of C_{k-1}.

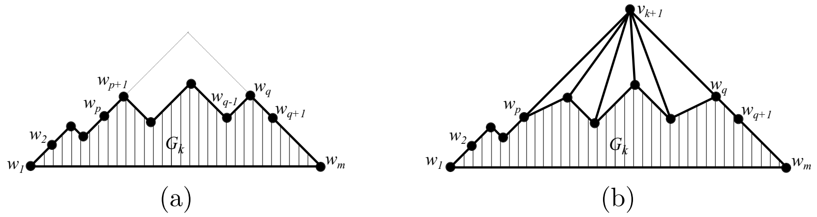
From Canonical Orderings to Straight-line Grid Drawings. The drawing algorithm proposed in [dPP90] inductively constructs a straight-line grid drawing of a maximal plane graph G relying on a canonical ordering of G. Let (u, v, w) be the outer face of G and let $v_1 = u, v_2 = v, v_3, \dots, v_n = w$ be a canonical ordering of G. Let G_k be the subgraph induced by $v_1, v_2, \cdots, v_k$. Then, de Fraysseix *et al.* prove there exists a drawing of G_k such that (refer to Fig. 2.4 (a)):

1. Vertex v_1 is at $(0, 0)$, vertex v_2 is at $(2k - 4, 0)$.

2. If $w_1 = v_1, w_2, \dots, w_m = v_2$ denote the vertices on the outer face of G_k (in order of their appearance) and $x(w_i)$ denotes the x-coordinate of w_i, then $x(w_1) < x(w_2) < \dots < x(w_m)$.

3. The edges (w_i, w_{i+1}), with $1 \leq i \leq m$, have slopes $\pi/4$ radiants or $-\pi/4$ radiants.

In order to maintain invariants (1), (2), and (3) when a new vertex v_{k+1} is added to the drawing, some vertices have to be shifted from their previous positions along the horizontal line they lie on. However, de Fraysseix *et al.* elegantly define which are the vertices that have to be moved in order to insert v_{k+1} still maintaining a planar drawing. In fact, they inductively assume that at step k of the algorithm, for each vertex $w_1, w_2, \dots, w_m$, a subset $M(k, w_i) \subseteq V(G_k)$ is defined such that:



(a) (b)

Figure 2.4: (a) Invariants of the drawing algorithm of de Fraysseix *et al.* (b) Incremental construction of the drawing of G_{k+1}.

- (a) $w_j \in M(k, w_i)$ if and only if $j \geq i$;
- (b) $M(k, w_1) \supset M(k, w_2) \supset \ldots \supset M(k, w_m)$;
- (c) For any non-negative numbers $\alpha_1, \alpha_2, \ldots, \alpha_m$, if all the vertices in $M(k, w_i)$ are moved by $\sum_{j=1}^{i} \alpha_j$, then the drawing remains straight-line and planar.

Invariants (1), (2), and (3) and properties (a), (b), and (c) are initially satisfied by setting $w_1 = (0, 0)$, $w_2 = (1, 1)$, $w_3 = (2, 0)$, $M(3, w_1) = \{w_1, w_2, w_3\}$, $M(3, w_2) = \{w_2, w_3\}$, and $M(3, w_3) = \{w_3\}$. When a new vertex v_{k+1} is inserted in the drawing (see Fig. 2.4 (b)), property (c) of sets $M(k, w_i)$ is applied with $\alpha_{p+1} = \alpha_q = 1$ and all other α_i equal to 0, where w_p and w_q are the first and the last neighbor of v_{k+1} on C_k. The new border of G_{k+1} is then $w_1, w_2, \ldots, w_p, v_{k+1}, w_q, \ldots, w_m$, and sets $M(k+1, w_i)$ are then defined so that $M(k+1, w_i) = M(k, w_i) \cup \{v_{k+1}\}$, for $i \leq p$, $M(k+1, v_{k+1}) = M(k, w_{p+1}) \cup \{v_{k+1}\}$, and $M(k+1, w_i) = M(k, w_i)$, for $i \geq q$. de Fraysseix *et al.* show how this modifications allow the partially constructed drawing to satisfy invariants (1), (2), and (3), and allow sets $M(k+1, w_i)$ to satisfy properties (a), (b), and (c).

When $k = n$ the resulting drawing is a planar straight-line drawing of G. Further, the width of the drawing is $2n - 4$, while its height is half of its width, hence $n - 2$.

Theorem 2.2 *(de Fraysseix, Pach, and Pollack [dPP90]) Any plane graph with n vertices has a straight-line planar drawing on the $(2n-4) \times (n-2)$ grid.*

It is worth noting that the described algorithm has been proposed by de Fraysseix *et al.* together with an $O(n \log n)$-time implementation. The authors conjectured that its complexity could be improved to $O(n)$. This bound was in fact achieved a few years later by Chrobak and Payne in [CP95].

Schnyder's Algorithm

In this section we describe the algorithm presented by Schnyder in [Sch90]. The approach of Schnyder is totally different from the one of de Fraysseix *et al.* [dPP90]. In fact, Schnyder's algorithm constructs the drawing by determining the coordinates of all the vertices in one shot. The algorithm relies on some results concerning planar graph embeddings that are indeed less intuitive than the canonical ordering of a plane graph used by de Fraysseix *et al.* Below, the main ideas behind Schnyder's algorithm are sketched.

Barycentric Representations and Straight-line Embeddings. Schnyder defines a *barycentric representation* of a graph G to be an injective function $v \in V(G) \rightarrow (x(v), y(v), z(v))$ that satisfies the following conditions:

1. $x(v) + y(v) + z(v) = 1$, for all vertices $v \in V(G)$.
2. For each edge $(u, v) \in E(G)$ and each vertex $w \notin \{u, v\}$, $x(u) < x(w)$ and $x(v) < x(w)$ hold, or $y(u) < y(w)$ and $y(v) < y(w)$ hold, or $z(u) < z(w)$ and $z(v) < z(w)$ hold.

Schnyder proves that, given any graph G, given any barycentric representation $v \rightarrow (x(v), y(v), z(v))$ of G, and given any three non-collinear points α, β, and γ in the three-dimensional space, the mapping $f : v \in V(G) \rightarrow v_1\alpha + v_2\beta + v_3\gamma$ is a straight-line planar embedding of G in the plane spanned by α, β, and γ.

Schnyder's Realizers. Schnyder proves that the interior edges of a plane graph G can be oriented and partitioned in three sets T_1, T_2, and T_3, so that the following conditions are satisfied:

1. The set of edges in T_i, for each $i = 1, 2, 3$, is a tree spanning all the internal vertices of G and exactly one external vertex; all the edges of T_i are directed towards this external vertex, which is the root of T_i; the external vertices belonging to T_1, to T_2, and to T_3 are distinct and appear in counter-clockwise order on the border of the outer face of G.
2. For each internal vertex v of G, v has outdegree one in each of T_1, T_2, and T_3; further, the counter-clockwise order of the edges incident to v is: leaving T_1, entering T_3, leaving T_2, entering T_1, leaving T_3, and entering T_2.

Schnyder calls a *realizer* of G the described partition of the edges of G. Fig. 2.5 (a) illustrates a realizer for a plane graph.

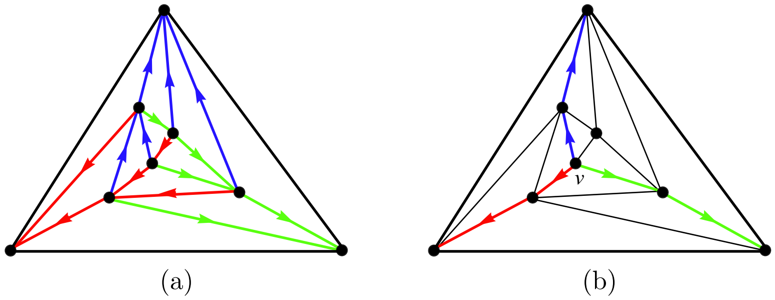


Figure 2.5: (a) A realizer for a plane graph. (b) Paths $P_1(v)$, $P_2(v)$, and $P_3(v)$ and regions $R_1(v)$, $R_2(v)$, and $R_3(v)$.

Given any realizer of a plane graph G, to each internal vertex $v \in V(G)$ three paths $P_i(v)$, with $i = 1, 2, 3$, are univocally associated so that $P_i(v)$ is the only path composed entirely of edges of T_i connecting v to the root of T_i (see Fig. 2.5 (b)). Such paths have only v as common vertex. Hence, $P_1(v)$, $P_2(v)$, and $P_3(v)$ divide G into three regions $R_1(v)$, $R_2(v)$, and $R_3(v)$, where $R_i(v)$ denotes the closed region opposite to the root of T_i.

From Realizers to Barycentric Representations. Schnyder describes two methods for obtaining a barycentric representation of a plane graph G starting from a realizer of G. In both methods, the coordinates of the barycentric representation are obtained by considering regions $R_1(v)$, $R_2(v)$, and $R_3(v)$ associated with each vertex v.

In the first method, for each vertex $v \in V(G)$, coordinates $x(v)$, $y(v)$, and $z(v)$ are set equal to the *number of the faces* in regions $R_1(v)$, $R_2(v)$, and $R_3(v)$, respectively, each divided by $2n - 5$. The assignment satisfies the first condition of a barycentric representation. In fact, the number of internal faces of a maximal plane graph G is equal to $2n - 5$, thus the sum of the number of faces in regions $R_1(v)$, $R_2(v)$, and $R_3(v)$ is equal to $2n - 5$, and hence $x(v)+y(v)+z(v) = 1$ holds for every vertex $v \in V(G)$. Further, the assignment satisfies the second condition of a barycentric representation, namely given any edge (u, v) and given any vertex $w \neq u, v$, vertices u and v both belong to a region $R_i(w)$, for some $i \in \{1, 2, 3\}$, hence they both have the corresponding coordinate smaller than the same coordinate of w. Then, in order to obtain a planar straight-line drawing of G, it is sufficient to apply the correspondence between barycentric representations and straight-line embeddings described at the beginning of this section, by setting $\alpha = (2n-5, 0)$, $\beta = (0, 2n-5)$, and $\gamma = (0, 0)$. The area of the resulting drawing is then $(2n-5) \times (2n-5)$.

In the second method, a definition of *weak barycentric representation* is given, which is similar to the one of barycentric representation, apart from the fact that the second condition is slightly relaxed allowing some equalities between the coordinates assigned to the vertices. Again, Schnyder shows that to any weak barycentric representation, a straight-line planar embedding in the plane spanned by any three points α, β, and γ corresponds. Then, for any given plane graph G, Schnyder shows how to obtain a weak barycentric representation by assigning to each vertex v coordinates which are equal to the *number of vertices* in regions $R_1(v)$, $R_2(v)$, and $R_3(v)$, where each region $R_i(v)$ is here meant to include the delimiting path $P_{i+1}(v)$, and to exclude the delimiting path $P_{i-1}(v)$. Then, in order to obtain a planar straight-line drawing of G, it is sufficient to apply the correspondence between weak barycentric representations and straight-line embeddings, by setting $\alpha = (n-2, 0)$, $\beta = (0, n-2)$, and $\gamma = (0, 0)$. The area of the resulting drawing is then $(n-2) \times (n-2)$.

Theorem 2.3 *(Schnyder [Sch90]) Any plane graph with n vertices has a straight-line planar drawing on the $(n-2) \times (n-2)$ grid.*

Lower Bounds

The quadratic area upper bound achieved by de Fraysseix, Pach, and Pollack's algorithm [dPP90] and by the Schynder's algorithm [Sch90] is asymptotically the best possible bound for straight-line drawings of planar graphs. In fact, almost ten years before the publication of such algorithms, Valiant observed in [Val81] that there exist n-vertex plane graphs requiring $\Omega(n^2)$ area in any straight-line planar drawing (in fact, in every poly-line planar drawing). The graph shown by Valiant to achieve such a bound is depicted in Fig. 2.6 (a). An improved lower bound was then shown by de Fraysseix, Pach, and Pollack in [dPP90]. In fact, they exhibited a class of graphs, known as *nested triangles graphs*, that is composed of a sequence of $n/3$ cycles, each of three vertices, nested one inside the other. A nested triangles graph with n vertices is easily shown to require $\left(\frac{2n}{3} - 1\right) \times \left(\frac{2n}{3} - 1\right)$ area in any grid drawing. A nested triangle graph is depicted in Fig. 2.6 (b).

The $\left(\frac{2n}{3} - 1\right) \times \left(\frac{2n}{3} - 1\right) = \frac{4n^2}{9} - \frac{4n}{3} + 1$ lower bound of de Fraysseix, Pach, and Pollack was improved only very recently by Frati and Patrignani [FP07].

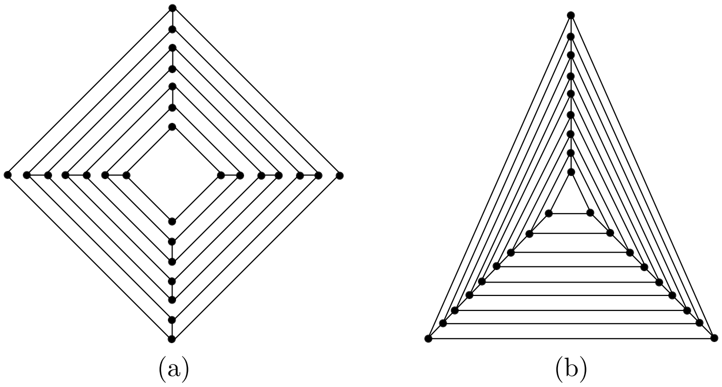


Figure 2.6: (a) The graph shown in [Val81] to prove a quadratic lower bound for the area requirements of plane graphs. (b) The graph shown in [dPP90] to prove a $\left(\frac{2n}{3}-1\right) \times \left(\frac{2n}{3}-1\right)$ lower bound for the area requirements of plane graphs.

In fact, they exhibited a plane graph requiring $\frac{4n^2}{9} - \frac{2n}{3}$ area in any straight-line drawing. The structure of such a graph is the same as the ones of the graphs shown by Valiant and by de Fraysseix, Pach, and Pollack. Namely, a linear number of cycles are nested one inside the other. However, the improvement comes from a 6-vertex graph which requires more than $\left(\frac{2n}{3}-1\right) \times \left(\frac{2n}{3}-1\right) = 3 \times 3$ area in any straight-line drawing. In fact, it can be proved by case analysis that the smallest grid drawings of the graph G_6 shown in Fig. 2.7 (a) require 2×7 and 3×4 area. Then, as shown in Fig. 2.7 (b), it is sufficient to nest a sequence of $n/3 - 2$ cycles, each of three vertices, outside G_6 to get the claimed $\frac{4n^2}{9} - \frac{2n}{3}$ lower bound.

Frati and Patrignani also observed that a quadratic area lower bound holds even for *planar graphs*. In fact, in any plane embedding of an n-vertex nested triangles graph (notice that in this setting the outer face can be suitably chosen), a sequence of $n/6$ cycles, each of three vertices, can be found, thus leading to a $\frac{n^2}{9} + \Omega(n)$ area lower bound. The same authours also observed that nested triangles graphs actually admit straight-line drawings in $\frac{2n^2}{9} + O(n)$ area, and conjectured that the latter bound is tight.

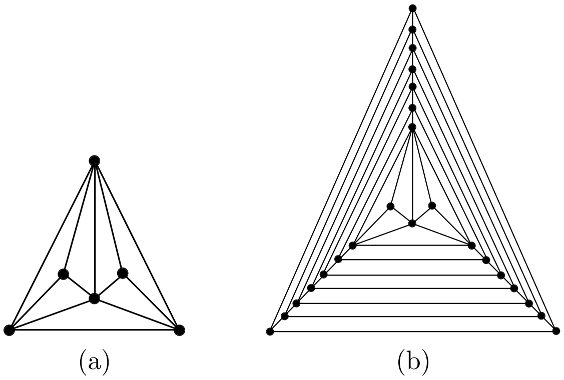


Figure 2.7: (a) Graph G_6 requiring 2×7 and 3×4 area in its smallest straight-line grid drawings. (b) A plane graph requiring $4n^2/9 - 2n/3$ area.

2.3 Poly-line Drawings

In this section, we expose the state of the art on poly-line drawings of planar graphs in small area. More results on poly-line drawings of planar graphs (and their subclasses) in small area are exhibited in Chapters 4, 7, and 9.

The first quadratic upper bound for the problem was shown by Woods in [Woo82]. Namely, the author proved that every planar graph has a planar poly-line drawing in $O(n^2)$ area and with $O(n^2)$ bends.

Very soon, the bound was improved by Di Battista and Tamassia, who showed that every plane graph has a poly-line drawing on a grid of size $(2n-5) \times (n-1)$, with a total number of $\frac{1}{3}(10n-31)$ bends. The result of Di Battista and Tamassia appeared in [DT88] and used as a main tool the following technique that has been later used in several other papers in which the construction of poly-line drawings is required: First, a grid visibility representation of the input graph is constructed. In a grid visibility representation, each vertex u is represented by a segment lying on an integer horizontal line, and each edge is represented by a segment lying on an integer vertical line. Such a representation can be constructed with the algorithm presented by Tamassia and Tollis in [TT86]. Second, a poly-line drawing is constructed from the visibility representation as follows: (i) replace each vertex-segment $\sigma(u)$ with a grid point $P(v)$ on $\sigma(u)$; (ii) replace each edge-segment spanning one vertical unit with a segment between the points $P(u)$ and $P(v)$, and replace each edge-segment spanning more than one vertical unit with a polygonal line connecting points $P(u)$, $(x(u, v), y(u) + 1)$, $(x(u, v), y(v) - 1)$, and $P(v)$.

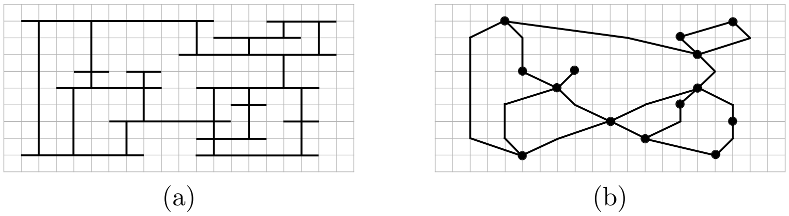


Figure 2.8: (a) A visibility representation Σ of a plane graph. (b) The poly-line drawing obtained from Σ by means of the algorithm of Di Battista and Tamassia [DT88].

Kant and He [KH93] proved that, for a 4-connected plane graph, the algorithm of Tamassia and Tollis produces visibility representations with $(n - 1) \times (n - 1)$ area. Kant [Kan97] showed how to compose the drawings of the 4-connected components of a general maximal plane graph G, in order to obtain a visibility representation of G with $(n-1) \times (n-1)$ area. Clearly, these upper bounds also apply to poly-line drawings, due to the just described Di Battista and Tamassia's algorithm [DT88]. However, a better bound is achieved by the already described Schnyder's algorithm (see [Sch90] and Sect. 2.2), since a straight-line drawing is a special case of poly-line drawing. Algorithms for constructing poly-line drawings with a good angular resolution in quadratic area were presented by Kant in [Kan96], by Gutwenger and Mutzel in [GM98], and by Cheng, Duncan, Goodrich and Kobourov in [CDGK01].

Finally, Bonichon, Le Saëc, and Mosbah proved in [BSM02] that every plane graph has a poly-line drawing in $(n - \lfloor \frac{p}{2} \rfloor - 1) \times (p + 1)$ area, where $p \leq \frac{2n-5}{3}$. Such an area bound is optimal up to the constants. In fact, the bound reads as $\frac{4(n-1)^2}{9}$ in the worst case, which matches the lower bound provided by the nested triangles graph (see Sect. 2.2). The algorithm of Bonichon *et al.* has the further appealing feature of constructing drawings with at most $n - 2$ bends, and with at most one bend per edge. The algorithm of Bonichon *et al.* uses as a main tool the Schnyder realizers (see Sect. 2.2).

2.4 Convex Drawings

In this section, we expose the state of the art on convex and strictly-convex drawings of planar graphs.

While every plane graph admits a planar straight-line drawing [Wag36, Far48, Ste51], not every plane graph admits a convex drawing. Tutte showed that every triconnected plane graph G admits a strictly-convex drawing with its outer face drawn as an arbitrary strictly-convex polygon P [Tut60]. Its algorithm starts by drawing the $f(G)$ as P; then, any other vertex is placed at the barycenter of the positions of its adjacent vertices. This results in a set of linear equations, namely, for each vertex $u \in V(G)$, denoting by E_u the set of edges incident to u in G:

$$x(u) = \frac{1}{|E_u|} \sum_{v \in E_u} x(v),$$

$$y(u) = \frac{1}{|E_u|} \sum_{v \in E_u} y(v).$$

Such a set of equations always admits a unique solution.

Characterizations of the graphs admitting a convex drawing were given by Tutte in [Tut60, Tut63], by Thomassen in [Tho80, Tho84], by Chiba, Yamanouchi, and Nishizeki in [CYN84], by Nishizeki and Chiba in [NC88], by Di Battista, Tamassia, and Vismara in [DTV01]. Chiba, Yamanouchi, and Nishizeki presented in [CYN84] a linear-time algorithm for producing convex drawings.

The area requirements of convex and strictly-convex grid drawings have been widely studied, especially for triconnected plane graphs.

Convex Drawings. Convex grid drawings of triconnected plane graphs can be realized on a quadratic-size grid. This was first shown by Kant in [Kan96]. In fact, Kant proved that such drawings can always be realized on a $(2n-4) \times (n-2)$ grid. The result is achieved by means of an incremental construction of a partial drawing in which, at the $(k+1)$-th step of the construction, either the drawing of a single vertex v_k or the drawing of a path $(v_{k,1}, v_{k,2}, \dots, v_{k,l})$ are added to the drawing of the partially constructed graph G_k. In the case in which v_k is added to the drawing, such a vertex could have several neighbors on the border of the outer face of G_k. In the case in which $(v_{k,1}, v_{k,2}, \dots, v_{k,l})$ is added to the drawing (see Fig. 2.9), such a path delimits the border of one internal face of G_{k+1}, together with a path on the border of the outer face of G_k, and all the vertices $v_{k,1}, v_{k,2}, \dots, v_{k,l}$ have degree exactly two in G_{k+1}.

This generalization of the canonical ordering allows to deal with plane graphs containing non-triangular faces.

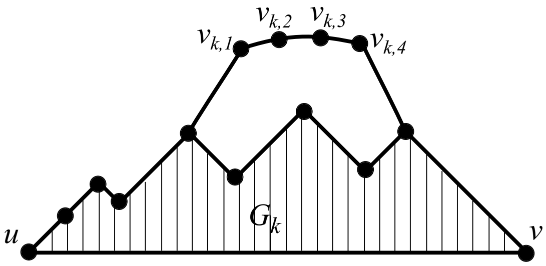


Figure 2.9: An illustration of the canonical ordering of a triconnected plane graph.

The bound of Kant was later improved down to $(n-2)\times(n-2)$ by Chrobak and Kant [CK97], and independently by Schnyder and Trotter [ST92]. The result of Chrobak and Kant again relies on the canonical ordering of a graph. On the other hand, the result of Schnyder and Trotter relies on a generalization of the Schnyder realizers in order to deal with triconnected plane graphs. Such an extension was independently shown by Di Battista, Tamassia, and Vismara [DTL99], who proved that every triconnected plane graph has a convex drawing on a $(f-2)\times(f-2)$ grid, where f is the number of faces of the graph. The best bound is currently, as far as we know, an $(n-2-\Delta)\times(n-2-\Delta)$ bound achieved by Bonichon, Felsner, and Mosbah in [BFM07]. The bound is again achieved by means of the use of the Schnyder realizers. The parameter Δ is dependent of the Schnyder realizers, and can variate among 0 and $\frac{n}{2}-2$.

Strictly-Convex Drawings. Strictly-convex drawings of triconnected plane graphs require cubic area in the worst case. In fact, a n-vertex cycle needs $\Omega(n^3)$ area in any grid realization (see, e.g., [And63, BP92, BT04]). The currently best lower bound, which has been proved by Rabinowitz in [Rab93], is $A(n) > \frac{n^3}{8\pi^2}$. The first polynomial upper bound for strictly-convex drawings of triconnected plane graphs has been proved by Chrobak, Goodrich, and Tamassia in [CGT96]. The authors showed that every triconnected plane graph admits a strictly-convex drawing in a $O(n^3)\times O(n^3)$ grid. Their idea consists in first constructing a (non-strictly-) convex drawing of the input graph, and then perturbing the positions of the vertices in order to achieve strict convexity. A more elaborated technique relying on the same idea allowed Rote to achieve an $O(n^{7/3}) \times O(n^{7/3})$ area upper bound in [Rot05], which was further improved by Barany and Rote to $O(n^2) \times O(n^2)$ in [BR06]. The last one is, as far as we know, the best known upper bound. One of the main differences between the Chrobak *et al.*'s algorithm, and the Rote's ones is that the former one constructs the intermediate non-strictly-convex drawing by making use of a canonical ordering of the graph, while the latter ones by making use of the Schnyder realizers.

2.5 Proximity Drawings

In this section, we expose the state of the art on proximity drawings of planar graphs. More results on proximity drawings of planar graphs (and their subclasses) in small area are exhibited in Chapters 3 and 8.

Given a set P of points in the plane, the *Delaunay triangulation* [Del34] of P is the internally-triangulated plane graph such that the three vertices of every face span a circumference which contains no point of P in its interior (see Fig. 2.10 (a)). Given a set P of points in the plane, a *Gabriel graph* [GS69] of P is obtained by connecting every two points u and v such that the disk having u and v as antipodal points does not contain any other point in its interior (see Fig. 2.10 (b)). Given a set P of points in the plane, a *minimum spanning tree* of P is defined as a tree having a vertex for each point of P and having minimum total edge length (see Fig. 2.10 (c)). Delaunay triangulations, Gabriel graphs, and minimum spanning trees are examples of *proximity graphs*, that is, graphs that can be constructed from given point sets so that the graph has a vertex for each point of the set, and there is an edge between two vertices if the corresponding points satisfy some proximity properties.

From a Graph Drawing point of view, it is a fundamental to characterize the class of proximity graphs, for a given definition of proximity. However, it turns out that characterizing the graphs that admit a proximity drawing, for a certain definition of proximity, is a difficult problem. For example, despite several research efforts (see, e.g., [Dil90, LL96, DS96]), characterizing the graphs that admit a *realization* (word which often substitutes *drawing* in the context of proximity graphs) as Delaunay triangulations is still an intriguing open problem. Further, the decision version of several realizability problems (that is, given a graph G and a definition of proximity, can G be realized as a proximity graph?) is $\mathcal{N}P$-hard. For example, Eades and Whitesides proved that deciding whether a tree can be realized as a minimum spanning tree is an $\mathcal{N}P$-hard problem [EW96b], and that deciding whether a graph can be realized

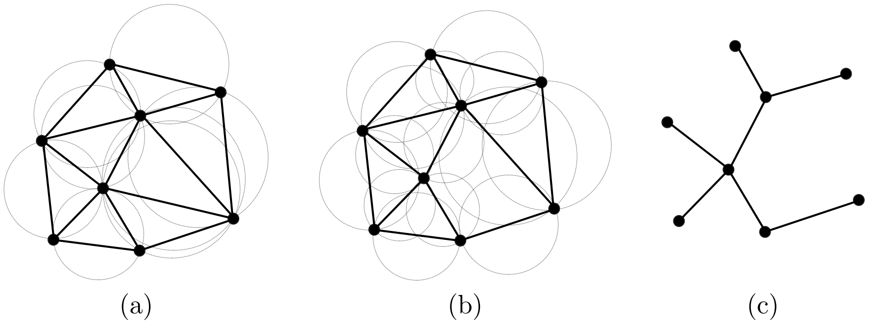


Figure 2.10: A set of points P and its (a) Delaunay triangulation, (b) Gabriel graph, and (c) minimum spanning tree. Notice that the minimum spanning tree is a subgraph of the Gabriel graph, which in turn is a subgraph of the Delaunay triangulation.

as a *nearest neighbor graph* is an $\mathcal{N}P$-hard problem [EW96a], as well. On the other hand, for several definitions of proximity graphs (such as Gabriel graphs and *relative neighborhood graphs*), the realizability problem is polynomial-time solvable for trees, as shown by Bose, Lenhart, and Liotta [BLL96]; further, Lubiw and Sleumer proved that maximal outerplanar graphs can be realized as *relative neighborhood graphs* and *Gabriel graphs* [LS93], result later extended by Lenhart and Liotta to all biconnected outerplanar graphs [LL96]. For more results about proximity drawings, see [DLL94, Lio95].

Most of the known algorithms to construct proximity drawings produce representations whose size increases exponentially with the number of vertices (see, e.g., [LS93, BLL96, LL96, DLW06]). However, an exponential-area lower bound was proved by Liotta, Tamassia, Tollis, and Vocca in [LTTV97] for the area requirements of Gabriel drawings. On the other hand, Penna and Vocca proved several polynomial upper bound for Gabriel drawings of trees in two and three dimensions [PV04]. The problem of constructing proximity drawings of graphs within polynomial area is considered as very challenging by several authors [MS92, BLL96, EW96b]. In particular, it is a long-standing open question whether trees with maximum degree 5 always admit realizations in polynomial area. Monma and Suri [MS92] conjectured that exponential area is sometimes required; however, Frati and Kaufmann proved in [MS92] that trees with maximum degree 4 always admit realizations in $O(n^{21.252})$ area.

2.6 Conclusions and Open Problems

In this chapter we have provided a panorama of the state of the art concerning minimum area drawings of plane and planar graphs, under several different drawing styles.

Concerning straight-line drawings, the best known upper bound is $n^2 + O(n)$[1], due to Schnyder [Sch90], while the best known lower bound is $\frac{4n^2}{9} + \Omega(n)$ [dPP90, FP07]. The following is hence a natural open question:

Open Problem 2.1 *Which are the tight bounds (up to a linear factor) for the area requirements of straight-line planar drawings of plane graphs?*

The same question is indeed interesting also for planar graphs, where the gap between the best upper bound (still the $n^2 + O(n)$ by Schnyder [Sch90]) and the best lower bound (that is only $\frac{n^2}{9} + \Omega(n)$ [FP07] is even wider.

Open Problem 2.2 *Which are the tight bounds (up to a linear factor) for the area requirements of straight-line planar drawings of planar graphs?*

However, in [FP07] the lower bound for such area requirements is conjectured to be at least $\frac{2n^2}{9} + \Omega(n)$.

Conjecture 2.1 *There exist planar graphs requiring $\frac{2n^2}{9} + \Omega(n)$ area in any straight-line drawing.*

Several problems on the area requirements of planar graphs remain open also for the other drawing styles that have been presented in the chapter. We cite here the one we find more appealing:

Open Problem 2.3 *Which are the asymptotic bounds for the area requirements of strictly-convex drawings of triconnected plane graphs?*

[1] At the *Topological & Geometric Graph Theory* International Conference, a four pages abstract was presented by Brandenburg entitled "Drawing Planar Graphs on $\frac{8}{9}n^2$ Area" [Bra08]. The paper sketches an algorithm, mainly based on the de Fraysseix, Pach, and Pollack's method, which constructs straight-line drawings of planar graphs in $\frac{4n}{3} \times \frac{2n}{3}$ area. Even if this bound, which clearly improves Schnyder's one, is now the best known bound for constructing straight-line drawings of planar graphs on a grid, we choose to not present the results of Brandenburg's paper in the state of the art discussed in this section. In fact, the paper has not yet, as far as we know, been published in an extended version, and hence we are not aware of (and thus could not illustrate) all its new ideas that allow to improve the de Fraysseix, Pach, and Pollack's algorithm.

Chapter 3

Greedy Drawings of Planar Graphs

In this chapter[1], we study greedy drawings of planar graphs, a kind of proximity drawings that has recently attracted several research efforts. We show an algorithm to construct greedy drawings of every given triangulation. The algorithm relies on two main results. First, we show how to construct greedy drawings of a fairly simple class of graphs, called triangulated binary cactuses. Second, we show that every triangulation can be spanned by a triangulated binary cactus. Further, we discuss how to extend our techniques in order to prove that every triconnected planar graph admits a greedy drawing. Such a result, which proves a conjecture by Papadimitriou and Ratajczak, was independently shown by Leighton and Moitra.

3.1 Introduction

The standard Internet routing protocol is as follows: Each computer is univocally identified by an *IP-address*; IP-addresses are *aggregated*, i.e., computers that are topologically or geographically close in the network are assigned addresses with the same most significative bits; consequently, routers do not have

[1] The contents of this chapter are a joint work with Patrizio Angelini and Luca Grilli, to appear in [AFG08] and submitted to journal. Thanks to Tom Leighton and Ankur Moitra for providing us with their paper, for introducing us to Gao and Richter's results, and for helping us to clarify the relationship between our results, their own, and Gao and Richter's ones.

to know the route to each address in the network, but they maintain in their *routing tables* only informations on the route to take for reaching each set of aggregated addresses. Such an approach does not work in many wireless networks, such as *ad-hoc* and *sensor networks*, where the addresses that are assigned to nodes geographically or topologically close are not necessarily similar. Despite of their importance, no universally-accepted communication protocol exists for such wireless environments.

Geographic routing is a class of routing protocols in which nodes forward packets based on their geographic locations. Among such protocols, *geometric routing*, or *greedy routing*, has been well investigated, because it relies on a very simple strategy in which, in order to forward packets, each node has to know only local informations and, obviously, the destination address. In fact, in the greedy routing a node forwards packets to a neighbor that is *closer than itself* to the destination's geographic location. Different distance metrics define different meanings for the word "closer", and consequently define different routing algorithms for the packet delivery. The most used and studied metric is of course the *Euclidean distance*.

The efficiency of the geographic routing algorithms strongly relies on the geographic coordinates of the nodes. This is indeed a drawback of such routing algorithms, for the following reasons: (i) Nodes of the network have to know their locations, hence they have to be equipped with GPS devices, which are expensive and increase the energy consumption of the nodes; (ii) geographic coordinates are independent of the network obstructions, i.e. obstacles making the communication between two close nodes impossible, and, more in general, they are independent of the network topology; this could lead to situations in which the communication fails because a *void* has been reached, i.e., the packet has reached a node whose neighbors are all farther from the destination than the node itself.

A brilliant solution to the geographic routing weakness has been proposed by Rao, Papadimitriou, Shenker and Stoica, who in [RPSS03] proposed a scheme in which nodes decide *virtual coordinates* and then apply the standard geometric routing algorithm relying on such virtual locations rather than on the real geographic coordinates. Clearly, virtual coordinates need not to reflect the nodes actual positions, hence they can be suitably chosen to guarantee that the geometric routing algorithm delivers packets with high probability. It has been experimentally shown that such an approach strongly improves the reliability of geometric routing [RPSS03, PR05]. Further, it has been proved that virtual coordinates guarantee geometric routing to work for every connected topology when they can be chosen in the hyperbolic plane [Kle07], even if only a logarithmic number of bits are available to store the coordinates of each node [EG08]. Moreover, some easy modifications of the routing algorithm guarantee that Euclidean virtual coordinates can be chosen so that the packet delivery always succeeds [BCGG06], even if the coordinates need to be locally computed [BCGW07].

Subsequent to the publication of Rao *et al.* paper [RPSS03], an intense research effort has been devoted to determine on which network topologies the Euclidean geometric routing with virtual coordinates is guaranteed to work.

From a graph-theoretic point of view, the problem can be restated as follows: Which are the graphs that admit a *greedy embedding*, i.e., a straight-line drawing Γ in the plane such that, for every pair of nodes u and v, there exists a *distance-decreasing path* in Γ? A path $(v_0, v_1, \ldots, v_m)$ is distance-decreasing if $d(v_i, v_m) < d(v_{i-1}, v_m)$, for $i = 1, \ldots, m$. Notice that greedy drawings are a type of proximity drawings (see Sect. 2.5), since a greedy drawing must satisfy some proximity properties among the points to which the vertices are mapped to.

In [PR05] Papadimitriou and Ratajczak conjectured the following:

Conjecture 3.1 *(Papadimitriou and Ratajczak [PR05]) Every triconnected planar graph admits a greedy embedding.*

Papadimitriou and Ratajczak showed that $K_{k,5k+1}$ has no greedy embedding, for $k \geq 1$. As a consequence, both the triconnectivity and the planarity are necessary, because there exist planar non-triconnected graphs, such as $K_{2,11}$, and non-planar triconnected graphs, such as $K_{3,16}$, that do not admit any greedy embedding. Further, they observed that, if a graph G has a greedy embedding, then any graph containing G as a spanning subgraph has a greedy embedding, as well. It follows that Conjecture 3.1 extends to all graphs which are spanned by a triconnected planar graph. Related to such an observation, Papadimitriou and Ratajczak proved that every triconnected graph that does not have a $K_{3,3}$-minor has a triconnected planar spanning subgraph.

There are a few classes of triconnected planar graphs for which the conjecture is easily shown to be true, for example graphs with a *Hamiltonian path* and *Delaunay Triangulations*. At SODA 2008 [Dha08], Dhandapani proved the conjecture for the first non-trivial class of triconnected planar graphs, namely he showed that every *triangulation* admits a greedy embedding. Triangulations are clearly an important graph class to study, as also remarked by Papadimitriou and Ratajczak [PR05]. The proof of Dhandapani is probabilistic, namely the author proves that, for every given triangulation G, a *random Schnyder* *drawing* of G [Sch90] is greedy with positive probability; hence, there exists a greedy drawing of every triangulation. Although such a proof is elegant, relying at the same time on an old Combinatorial Geometry theorem, known as the *Knaster-Kuratowski-Mazurkievicz Theorem* [KKM29], and on standard Graph Drawing techniques, as the *Schnyder realizers* [Sch90] and the *canonical orderings* of a triangulation [dPP90], it does not lead to an embedding algorithm.

In this chapter we show an algorithm for constructing greedy drawings of triangulations. The algorithm relies on a different and maybe easier approach with respect to the one used by Dhandapani. We define a simple class of graphs, called *triangulated binary cactuses*, and we provide an algorithm to construct a greedy drawing of any such a graph. Further, we show how to find, for every triangulation, a triangulated binary cactus spanning it. It is clear that the previous statements imply an algorithm for constructing greedy drawings of triangulations. Namely, consider any triangulation G, apply the algorithm to find a triangulated binary cactus S spanning G, and then apply the algorithm to construct a greedy drawing of S. As already observed, adding edges to a greedy drawing leaves the drawing greedy, hence S can be augmented to G, obtaining the desired greedy drawing of G.

Theorem 3.1 *Given a triangulation G, there exists an algorithm to compute a greedy drawing of G.*

Further, we provide an algorithm to construct greedy drawings of general triconnected planar graphs. The strategy of such an algorithm is the same as the one of the algorithm for constructing greedy drawings of triangulations. In fact, we define a simple class of graphs, called *non-triangulated binary cactuses*, and we provide an algorithm to construct a greedy drawing of any such a graph. Finally, we show how to find, for every triconnected planar graph, a non-triangulated binary cactus spanning it. Such a result proves Conjecture 3.1; however, the conjecture has been very recently (and independently) proved by Leighton and Moitra [LM08], by using techniques which are amazingly similar to ours. Hence, we will only sketch how to modify the algorithm we provide for triangulations in order to make it work for general triconnected planar graphs.

The rest of the chapter is organized as follows. In Sect. 3.2 we introduce triangulated binary cactuses; in Sect. 3.3 we show an algorithm to construct greedy drawings of triangulated binary cactuses; in Sect. 3.4 we show an algorithm to construct a triangulated binary cactus spanning a given triangulation; in Sect. 3.5 we show how to modify the algorithm described for triangulations in order to make it work for general triconnected planar graphs; finally, in Sect. 3.6 we conclude and present some open problems.

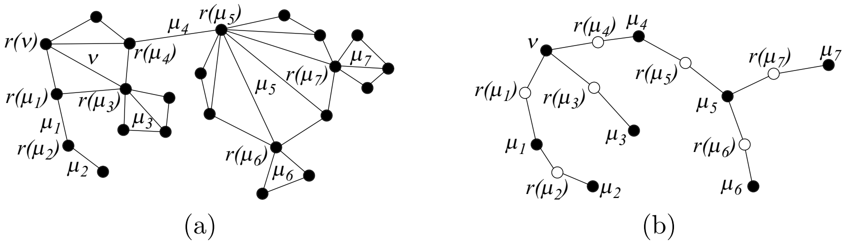
3.2 Triangulated Binary Cactuses

Consider a graph G. Consider its BC-tree $\mathcal{T}$ and suppose it is rooted at a specific B-node corresponding to a block ν. When the BC-tree $\mathcal{T}$ of a graph G is rooted at a certain block ν, we denote by $G(\mu)$ the subgraph of G induced by all the vertices in the blocks contained in the subtree of $\mathcal{T}$ rooted at μ. In a rooted BC-tree $\mathcal{T}$ of a graph G, for each B-node μ we denote by $r(\mu)$ the cutvertex of G parent of μ in $\mathcal{T}$. If μ is the root of $\mathcal{T}$, i.e., $\mu = \nu$, then we let $r(\mu)$ denote any non-cutvertex node of the block associated with μ. In the following, unless otherwise specified, each considered BC-tree is meant to be rooted at a certain B-node ν such that the block associated with ν has at least one vertex $r(\nu)$ which is not a cutvertex. It is not difficult to see that such a block exists in every planar graph.

A *triangulated binary cactus* S, in the following two sections simply called *binary cactus*, is a connected graph such that (see Fig 3.1):

- The block associated with each B-node of $\mathcal{T}$ is either an edge or a *triangulated cycle*, i.e., a cycle $(r(\mu), u_1, u_2, \ldots, u_h)$ triangulated by the edges from $r(\mu)$ to each of $u_1, u_2, \ldots, u_h$.

- Every cutvertex is shared by exactly two blocks of S.



(a) (b)

Figure 3.1: (a) A binary cactus S. (b) The block-cutvertex tree of S. White (resp. black) circles represent C-nodes (resp. B-nodes).

3.3 Greedy Drawings of Binary Cactuses

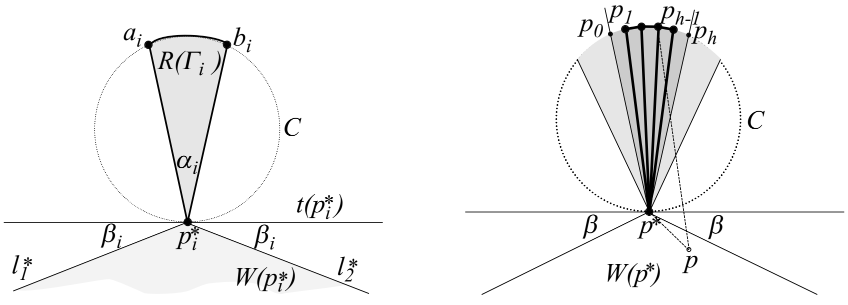
In this section, we give an algorithm to compute a greedy drawing of a binary cactus S. Such a drawing is constructed by a bottom-up traversal of the BC-tree $\mathcal{T}$ of S.

Consider the root μ of a subtree of $\mathcal{T}$ corresponding to a block of S, consider the k children of μ, which correspond to cutvertices of S, and consider the children of such cutvertices, say $\mu_1, \mu_2, \dots, \mu_k$. Notice that each C-node child of μ is parent of exactly one B-node μ_i of $\mathcal{T}$, by the definition of binary cactus. For each $i = 1, \dots, k$, inductively assume to have a drawing Γ_i of $S(\mu_i)$ satisfying the properties listed below.

Let C be a circumference, let (a_i, b_i) be an arc of C, let p_i^* be a point of C such that the diameter through p_i^* cuts (a_i, b_i) in two arcs of the same length. Let α_i and β_i be any two angles less than $\pi/4$ such that $\beta_i \geq \alpha_i$. Consider the tangent $t(p_i^*)$ to C in p_i^*. Consider two half-lines l_1^* and l_2^* incident to p_i^*, lying on the opposite part of C with respect to $t(p_i^*)$, and forming angles equal to β_i with $t(p_i^*)$. Denote by $W(p_i^*)$ the wedge centered at p_i^*, delimited by l_1^* and l_2^*, and not containing C. Refer to Fig. 3.2 (a).

- *Property 1.* Γ_i is a greedy drawing.
- *Property 2.* Γ_i is entirely contained inside a region $R(\Gamma_i)$ delimited by arc (a_i, b_i), and by segments $\overline{p_i^* a_i}$ and $\overline{p_i^* b_i}$. The angle $\widehat{a_i p_i^* b_i}$ is α_i.
- *Property 3.* For every vertex v in $S(\mu_i)$ and for every point p internal to $W(p_i^*)$, there exists in Γ_i a path $(v = v_0, v_1, \dots, v_l = r(\mu_i))$ from v to $r(\mu_i)$ such that $d(v_j, p) < d(v_{j-1}, p)$, for $j = 1, \dots, l$.
- *Property 4.* For every vertex v in $S(\mu_i)$ and for every point p internal to $W(p_i^*)$, $d(v, p_i^*) < d(v, p)$, for $j = 1, \dots, l$.

In the base case, block μ has no child. Denote by $(r(\mu) = u_0, u_1, \dots, u_{h-1})$ the block of S corresponding to μ. Notice that $h \geq 2$. Consider any circumference C with center c. Let p^* be the point of C with smallest y-coordinate. Consider the wedges $W(p^*, \alpha)$ and $W(p^*, \alpha/2)$ with angles α and $\alpha/2$, respectively, incident to p^* and such that the diameter of C through p^* is their bisector (see Fig. 3.2 (b)). Place $r(\mu)$ at p^*. Denote by p'_a and p'_b the intersection points (different from p^*) of the half-lines delimiting $W(p^*, \alpha/2)$ with C. Denote by A the arc of C between p'_a and p'_b and not containing p^*. Consider $h+1$ points $p_0, p_1, \dots, p_h$ on A such that $p_0 = p'_a$ and $p_h = p'_b$, and the distance between



(a)

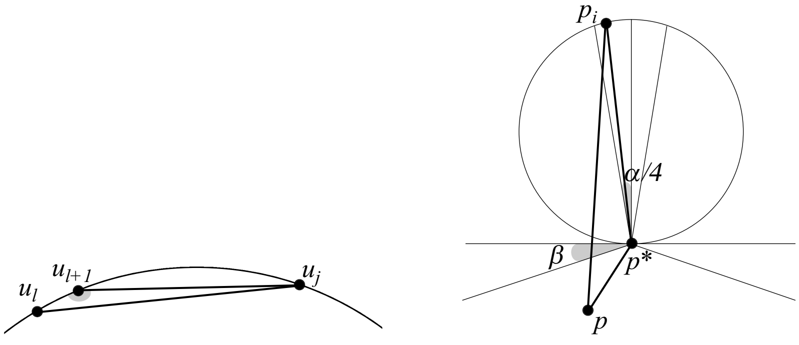
(b)

Figure 3.2: (a) Illustration for Properties 1–4 of Γ. (b) Base case of the algorithm. The light and dark shaded region represents $R(\Gamma)$ (the angle of $R(\Gamma)$ at p^* is α). The dark shaded region represents the intersection of $W(p^*, \alpha/2)$ with the circle delimited by C.

any two consecutive points p_i and p_{i+1} is the same. Place vertex u_i at point p_i, for $i = 1, 2, \ldots, h-1$. Notice that, if $h = 2$, μ corresponds to an edge of S that is drawn as a vertical segment, with u_1 above u_0.

In order to show that the constructed drawing Γ satisfies Property 1, consider any two vertices u_i and u_j, with $i < j$. If $i = 0$, then u_0 and u_j are joined by an edge, which provides a distance-decreasing path between them. Otherwise, we prove that $(u_i, u_{i+1}, \ldots, u_j)$ is a distance-decreasing path from u_i to u_j, the proof that $(u_j, u_{j-1}, \ldots, u_i)$ is a distance-decreasing path from u_j to u_i being analogous. For each $l = i, i+1, \ldots, j-2$, angle $\widehat{u_l u_{l+1} u_j}$ is greater than $\pi/2$, because triangle (u_l, u_{l+1}, u_j) is inscribed in less than half a circumference with u_{l+1} as middle point (see Fig. 3.3 (a)). Hence, (u_l, u_j) is the longest side of triangle (u_l, u_{l+1}, u_j) and $d(u_{l+1}, u_j) < d(u_l, u_j)$ follows. Drawing Γ satisfies Property 2 by construction. In order to prove that Γ satisfies Property 3, we have to prove that, for every vertex u_i, with $i \geq 1$, and for every point p in $W(p^*)$, $d(u_0, p) < d(u_i, p)$. Angle $\widehat{pp^*p_i}$ is at least $\beta + (\frac{\pi}{2} - \frac{\alpha}{4})$, which is more than $\pi/2$ (see Fig. 3.3 (b)). It follows that segment $\overline{pp_i}$ is the longest side of triangle (p, p^*, p_i), thus proving that $d(u_0, p) < d(u_i, p)$. For the same reason $d(u_0, u_i) < d(p, u_i)$, hence proving Property 4.

Now we discuss the inductive case, that is, suppose that μ is a node of $\mathcal{T}$ having k children. We show how to construct a drawing Γ of $S(\mu)$ satisfying Properties 1–4 with parameters α and β. Refer to Fig. 3.4. Denote by $(r(\mu) =$



(a) (b)

Figure 3.3: (a) Γ satisfies Property 1. (b) Γ satisfies Properties 3 and 4.

$u_0, u_1, \ldots, u_{h-1}$) the block of S corresponding to μ. Remember that $h \geq 2$ and that, if $h = 2$ the block is an edge, otherwise it is a triangulated cycle. Consider any circumference C with center c. Let p^* be the point of C with smallest y-coordinate. Consider the wedges $W(p^*, \alpha)$ and $W(p^*, \alpha/2)$ with angles α and $\alpha/2$, respectively, incident to p^* and such that the diameter of C through p^* is their bisector. Region $R(\Gamma)$ is the intersection region of $W(p^*, \alpha)$ with the closed circle delimited by C.

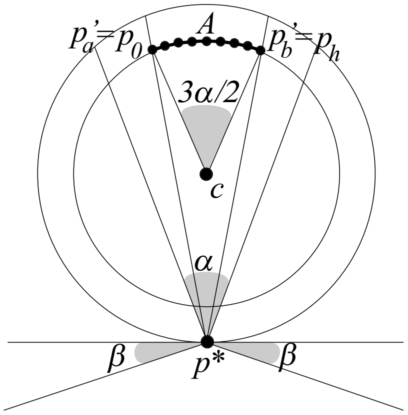


Figure 3.4: Construction of a drawing Γ in the inductive case of the algorithm.

Consider a circumference C' with center c intersecting the two lines delimiting $W(p^*, \alpha/2)$ in two points p'_a and p'_b such that angle $\widehat{p'_a c p'_b} = 3\alpha/2$. It is not difficult to see that such a circumference always exists. Denote by A the arc of C' delimited by p'_a and p'_b and farther from p^*. Consider $h+1$ points $p_0, p_1, \dots, p_h$ on A such that $p_0 = p'_a$ and $p_h = p'_b$, and the distance between any two consecutive points p_i and p_{i+1} is the same. Observe that, for each $i = 0, 1, \dots, h-1$, angle $\widehat{p_i c p_{i+1}} = \frac{3\alpha}{2h}$.

First, we draw the block of S corresponding to μ. As in the base case, place vertex $u_0 = r(\mu)$ at p^* and, for $i = 1, 2, \dots, h-1$, place u_i at point p_i. Recursively construct a drawing Γ_i of $S(\mu_i)$ satisfying Properties 1–4 with $\alpha_i = \frac{3\alpha}{16h}$ and $\beta_i = \frac{3\alpha}{8h}$.

We are going to place each drawing Γ_i of $S(\mu_i)$ together with the constructed drawing of the block of S corresponding to μ, thus obtaining a drawing Γ of $S(\mu)$. Notice that not all the h nodes u_i are cutvertices of S. However, with a slight abuse of notation, we suppose that block $S(\mu_i)$ has to be placed at node u_i. Refer to Fig 3.5. Consider point p_i and its "neighbors" p_{i-1} and p_{i+1}, for $i = 1, 2, \dots, h-1$. Consider lines $t(p_{i-1})$ and $t(p_{i+1})$ tangent to C' in p_{i-1} and p_{i+1}, respectively. Further, consider circumferences C_{i-1} and C_{i+1} centered at p_{i-1} and p_{i+1}, respectively, and passing through p_i. Moreover, consider lines h_{i-1} and h_{i+1} tangent to C_{i-1} and C_{i+1} in p_i, respectively. For each point p_i, with $i = 0, \dots, h$, consider two half-lines t^i_1 and t^i_2 incident to p_i, forming angles $\beta_i = \frac{3\alpha}{8h}$ with $t(p_i)$, and both lying in the half-plane delimited by $t(p_i)$ and containing C'. Denote by $W(p_i)$ the wedge delimited by t^i_1 and by t^i_2, and containing c.

We will place Γ_i inside (a part of) the bounded region R_i intersection of: (i) the half-plane H^{i-1} delimited by h_{i-1} and not containing C_{i-1}, (ii) the half-plane H^{i+1} delimited by h_{i+1} and not containing C_{i+1}, (iii) wedge $W(p_{i-1})$, (iv) wedge $W(p_{i+1})$, and (v) the circle delimited by C.

First, we prove that R_i is "large enough" to contain Γ_i, namely we claim that there exists an isosceles triangle T that has an angle larger than $\alpha_i = \frac{3\alpha}{16h}$ incident to p_i and that is completely contained in R_i. Such a triangle will have the further feature that the angle incident to p_i is bisected by the line l_i through c and p_i.

Lines h_{i-1} and h_{i+1} are both passing through p_i; we prove that they have different slopes and we compute the angles that they form at p_i. Refer to Fig. 3.6. Line h_{i-1} forms an angle of $\pi/2$ with segment $\overline{p_{i-1}p_i}$; angle $\widehat{cp_ip_{i-1}}$ is equal to $\frac{\pi}{2} - \frac{3\alpha}{4h}$, since $\widehat{p_i c p_{i-1}} = \frac{3\alpha}{2h}$ and since triangle (p_{i-1}, c, p_i) is isosceles. Hence, the angle delimited by h_{i-1} and l_i is $\pi - \pi/2 - (\frac{\pi}{2} - \frac{3\alpha}{4h}) = \frac{3\alpha}{4h}$. Anal-

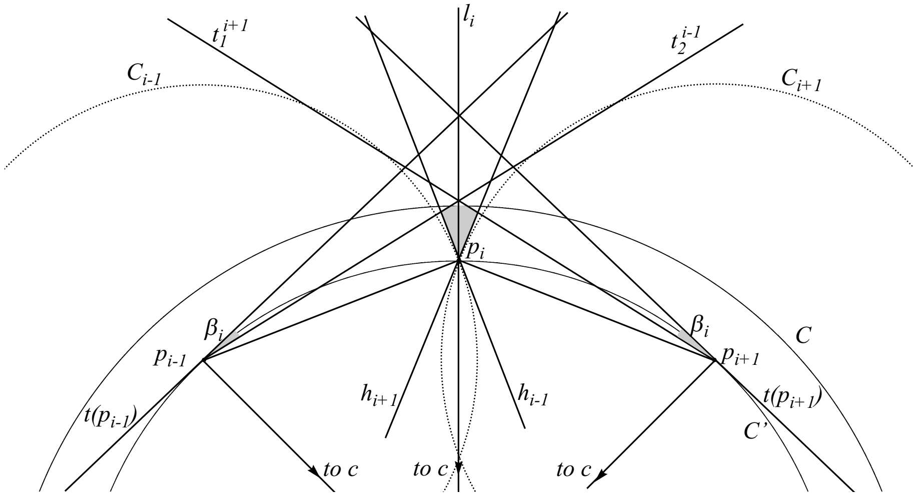


Figure 3.5: Lines and circumferences in the construction of Γ. The shaded areas represent angles β_i and region R_i.

ogously, the angle between l_i and h_{i+1} is $\frac{3\alpha}{4h}$. Hence, the intersection of H^{i-1} and H^{i+1} is a wedge $W(p_i, h_{i-1}, h_{i+1})$ centered at p_i, with an angle of $\frac{3\alpha}{2h}$, and bisected by l_i.

We claim that each of t_2^{i-1} and t_1^{i+1} cuts the border of $W(p_i, h_{i-1}, h_{i+1})$ twice. The angle between $t(p_{i-1})$ and $\overline{p_{i-1}p_i}$ is $\frac{3\alpha}{4h}$, namely the angle between $t(p_{i-1})$ and $\overline{cp_{i-1}}$ is $\pi/2$, and angle $\widehat{cp_{i-1}p_i}$ is $\frac{\pi}{2} - \frac{3\alpha}{4h}$. The angle between $t(p_{i-1})$ and t_2^{i-1} is $\beta_i = \frac{3\alpha}{8h}$, by construction. Hence, the angle between t_2^{i-1} and $\overline{p_{i-1}p_i}$ is $\frac{3\alpha}{4h} - \frac{3\alpha}{8h} = \frac{3\alpha}{8h}$. Since the slope of both h_{i-1} and h_{i+1} with respect to $\overline{p_{i-1}p_i}$ is greater than $\frac{3\alpha}{8h}$ and less than $\pi - \frac{3\alpha}{8h}$, namely the slope of h_{i-1} and h_{i+1} with respect to $\overline{p_{i-1}p_i}$ is $\frac{\pi}{2}$ and $\frac{\pi}{2} + \frac{3\alpha}{2h}$, respectively (notice that $\alpha \leq \pi/4$ and $h \geq 2$), then t_2^{i-1} intersects both h_{i-1} and h_{i+1}. It can be analogously proved that t_1^{i+1} intersects h_{i-1} and h_{i+1}. It follows that the intersection of H^{i-1}, H^{i+1}, $W(p_{i-1})$, and $W(p_{i+1})$ contains a triangle T as required by the claim (notice that the angle of T incident to p_i is $\frac{3\alpha}{2h}$). Considering circumference C does not invalidate the existence of T, since C is concentric with C' and has a bigger radius, hence T can always be chosen sufficiently small so that it completely lies inside C.

Now Γ_i can be placed inside T, by scaling Γ_i down till it fits inside T (see

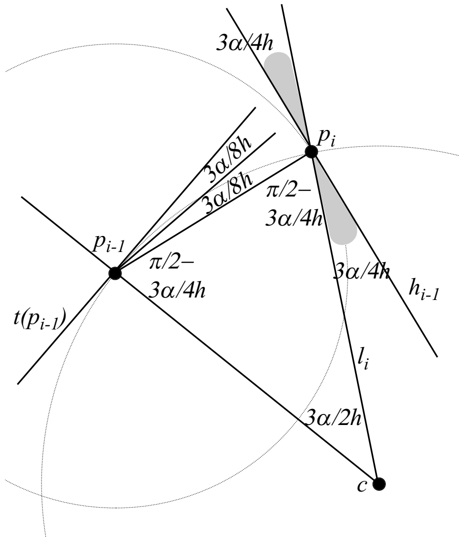


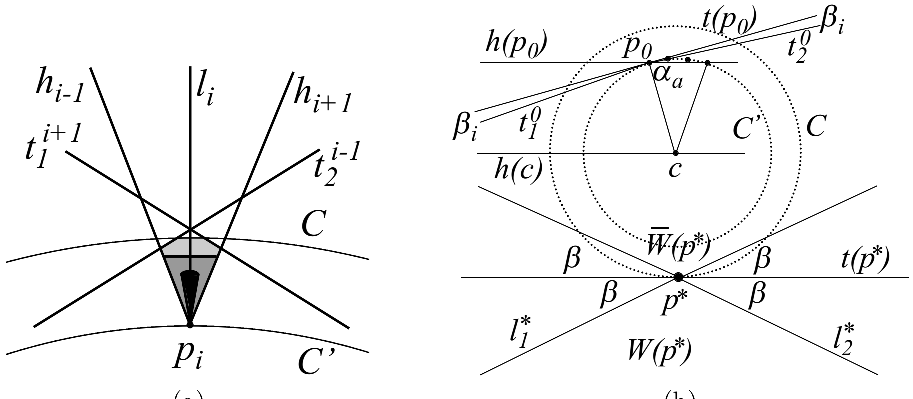
Figure 3.6: The angle between l_i and h_{i-1}.

Fig. 3.7 (a)). The scaling always allows to place Γ_i inside T, since the angle of $R(\Gamma_i)$ incident to p is $\alpha_i = \frac{3\alpha}{16h}$, that is smaller than the angle of T incident to p_i, which is $\frac{3\alpha}{2h}$. In particular, we choose to place Γ_i inside T so that l_i bisects the angle of $R(\Gamma_i)$ incident to p_i. This concludes the construction of Γ.

Consider the tangent $t(p^*)$ to C in p^*. Consider two half-lines l_1^* and l_2^* incident to p^*, lying on the opposite part of C with respect to $t(p^*)$, and forming angles equal to β with $t(p^*)$. Denote by $W(p^*)$ the wedge centered at p^*, delimited by l_1^* and l_2^*, and not containing C. We have the following lemmata.

Lemma 3.1 *The closed wedge $W(p^*)$ is completely contained inside the open wedge $W(p_i)$, for each $i = 0, 1, \ldots, h$.*

Proof: Consider any point p_i. First, observe that p_i is contained in the wedge $\overline{W}(p^*)$ obtained by reflecting $W(p^*)$ with respect to $t(p^*)$. Namely, p_i is contained in $W(p^*, \alpha/2)$, which is in turn contained inside $\overline{W}(p^*)$, since $\alpha/2 < \pi - 2\beta$, as a consequence of the fact that $\pi/4 > \beta \geq \alpha$. Hence, in order to prove the lemma, it suffices to show that the absolute value of the slope of each of t_1^i and t_2^i is less than the absolute value of the slope of the half-lines



(a) (b)

Figure 3.7: (a) Placement of Γ inside R_i. Region $R(\Gamma)$ is the darkest, triangle T is composed of $R(\Gamma)$ and of the second darkest region, R_i is composed of T and of the light shaded region. (b) Illustration for the proof of Lemma 3.1.

delimiting $W(p^*)$. Such latter half-lines form angles of β, by construction, with the x-axis.

The slope of t_1^i can be computed by summing up the slope of t_1^i with respect to $t(p_i)$ with the slope of $t(p_i)$. The former slope is equal to $\beta_i = \frac{3\alpha}{8h}$, by construction. Recalling that $t(p_i)$ is the tangent to A in p_i, the slope of $t(p_i)$ is bounded by the maximum among the slopes of the tangents to points of A. Such a maximum is clearly achieved at p_0 and p_h and is equal to $3\alpha/4$. Namely, refer to Fig. 3.7 (b) and consider the horizontal lines $h(c)$ and $h(p_0)$ through c and p_0, respectively, that are traversed by radius (c, p_0). Such a radius forms angles of $\pi/2$ with $t(p_0)$; hence, the slope of $t(p_0)$, that is equal to the angle between $t(p_0)$ and $h(p_0)$, is $\pi/2$ minus the angle α_a between $h(p_0)$ and (c, p_0). Angle α_a is the alternate interior of the angle between $h(c)$ and (c, p_0), which is complementary to the half of angle $\widehat{p_0 c p_h}$, which is equal to $3\alpha/2$, by construction. Hence, α_a is equal to $\frac{\pi}{2} - \frac{3\alpha}{4}$ and the slope of $t(p_0)$ is $\frac{3\alpha}{4}$.

It follows that the absolute value of the slope of t_1^i is at most $\frac{3\alpha}{4} + \frac{3\alpha}{8h}$, which is less than α, since $h \geq 2$, and hence less than β. Analogously, the absolute value of the slope of t_2^i is less than β, and the lemma follows. □

Corollary 3.1 *Point p^* is inside the open wedge $W(p_i)$, for each $i = 1, 2, \ldots, h$.*

Lemma 3.2 *For every pair of indices i and j such that $1 \leq i < j \leq k$, the drawing of $S(\mu_j)$ is contained inside $W(p_i)$ and the drawing of $S(\mu_i)$ is contained inside $W(p_j)$.*

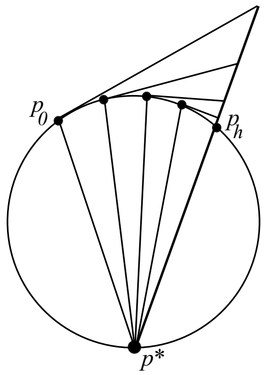


Figure 3.8: Illustration for the proof of Lemma 3.2.

Proof: We prove that the drawing of $S(\mu_j)$ is contained inside $W(p_i)$, the proof that the drawing of $S(\mu_i)$ is contained inside $W(p_j)$ being analogous. If $S(\mu_i)$ and $S(\mu_j)$ are consecutive, i.e., the cutvertices parents of $S(\mu_i)$ and $S(\mu_j)$ are u_i and u_j and $j = i + 1$, then the statement is true by construction. Suppose $S(\mu_i)$ and $S(\mu_j)$ are not consecutive. Refer to Fig. 3.8. Consider the triangle T_i delimited by (p^*, p_i), by t_2^i, and by the line through p^* and p_h. Such a triangle contains the triangle delimited by (p^*, p_{i+1}), by t_2^{i+1}, and by the line through p^* and p_h, which in turn contains the triangle delimited by (p^*, p_{i+2}), by t_2^{i+2}, and by the line through p^* and p'_b. The repetition of such an argument shows that T_i contains the triangle T_{j-1} delimited by (p^*, p_{j-1}), by t_2^{j-1}, and by the line through p^* and p_h. By construction, Γ_j lies inside T_{j-1}, and the lemma follows. □

We prove that the constructed drawing Γ satisfies Properties 1–4.

Property 1. We show that, for every pair of vertices w_1 and w_2, there exists a distance-decreasing path between w_1 and w_2 in Γ. If both w_1 and w_2 are internal to the same graph $S(\mu_i)$, the property follows by induction. If $w_2 = r(\mu)$ and w_1 is a node in $S(\mu_i)$, then, by Property 3, there exists a path $(w_1 = v_0, v_1, \ldots, v_l = r(\mu_i))$ from w_1 to $r(\mu_i)$ such that, for every point p in $W(p_i)$, $d(v_j, p) < d(v_{j-1}, p)$, for $j = 1, 2, \dots, l$. By Corollary 3.1, p^* is contained inside $W(p_i)$. Hence, path $(w_1 = v_0, v_1, \dots, v_l = r(\mu_i), w_2 = r(\mu))$ is a distance-decreasing path between w_1 and w_2. If $w_1 = r(\mu)$ and w_2 is a node in $S(\mu_i)$, then, by induction, there exists a distance-decreasing path $(v_1 = r(\mu_i), v_2, \dots, v_l = w_2)$. By Corollary 3.1, p^* is contained inside $W(p_i)$. Hence, by Property 4, $d(p_i, w_2) < d(p^*, w_2)$. It follows that path $(w_1 = r(\mu), v_1 = r(\mu_i), v_2, \dots, v_l = w_2)$ is a distance-decreasing path between w_1 and w_2. If w_1 belongs to $S(\mu_i)$ and w_2 belongs to $S(\mu_j)$ then suppose, w.l.o.g., that $j > i$. We show the existence of a distance-decreasing path $\mathcal{P}$ in Γ, composed of three subpaths $\mathcal{P}_1, \mathcal{P}_2$, and $\mathcal{P}_3$. By Property 3, Γ_j is such that there exists a path $\mathcal{P}_1 = (w_1 = v_0, v_1, \dots, v_l = r(\mu_i))$ from w_1 to $r(\mu_i)$ such that, for every point p in $W(p_i)$, $d(v_j, p) < d(v_{j-1}, p)$, for $j = 1, 2, \dots, l$. By Lemma 3.2, drawing Γ_j, and hence vertex w_2, is contained inside $W(p_i)$. Hence, path $\mathcal{P}_1$ decreases the distance from w_2 at every vertex. Path $\mathcal{P}_2 = (u_i = r(\mu_i), u_{i+1}, \dots, u_j = r(\mu_j))$ is easily shown to decrease the distance from w_2 at every vertex. In fact, for each $l = i, i+1, \dots, j-2$, angle $\widehat{u_l u_{l+1} u_j}$ is greater than $\pi/2$, because triangle (u_l, u_{l+1}, u_j) is inscribed in less than half a circumference with u_{l+1} as middle point. Angle $\widehat{u_l u_{l+1} w_2}$ is strictly greater than $\widehat{u_l u_{l+1} u_j}$, hence it is the biggest angle in triangle (u_l, u_{l+1}, w_2), which implies $d(u_{l+1}, w_2) < d(u_l, w_2)$. By induction, there exists a distance-decreasing path $\mathcal{P}_3$ from $r(\mu_j)$ to w_2, thus obtaining a distance-decreasing path $\mathcal{P}$ from w_1 to w_2.

Property 2. Such a property holds for Γ by construction.

Property 3. Consider any node v in $S(\mu_i)$ and consider any point p internal to $W(p^*)$. By Lemma 3.1, p is internal to $W(p_i)$, as well. By induction, there exists a path $(v = v_0, v_1, \dots, v_l = r(\mu_i))$ such that $d(v_j, p) < d(v_{j-1}, p)$, for $j = 1, 2, \dots, l$. Hence, path $(v = v_0, v_1, \dots, v_l = r(\mu_i), v_{l+1} = r(\mu))$ is a path such that $d(v_j, p) < d(v_{j-1}, p)$, for $j = 1, 2, \dots, l+1$, if and only if $d(r(\mu), p) < d(r(\mu_i), p)$. Angle $\widehat{pp^*r(\mu_i)}$ is at least $\beta + (\frac{\pi}{2} - \frac{\alpha}{2})$, which is more than $\pi/2$. It follows that $(p, r(\mu_i))$ is the longest side of triangle $(p, p^*, r(\mu_i))$, thus proving that $d(p, p^*) < d(p, r(\mu_i))$ and that Property 3 holds for Γ.

Property 4. By Property 2, v is contained inside the wedge $W(p^*, \alpha)$ with angle α, centered at p^*, and bisected by the line through p^* and c. Consider any point p inside $W(p^*)$. Angle $\widehat{pp^*v}$ is at least $\beta + (\frac{\pi}{2} - \frac{\alpha}{2})$, which is more than $\pi/2$. It follows that (p, v) is the longest side of triangle

(p, p^*, v), thus proving that $d(p, v) < d(p^*, v)$ and that Property 4 holds for Γ.

When the induction is performed with μ equal to the root ν of the BC-tree $\mathcal{T}$, we obtain a greedy drawing of S, thus proving the following:

Theorem 3.2 *There exists an algorithm that constructs a greedy drawing of any triangulated binary cactus.*

3.4 Spanning a Triangulation with a Binary Cactus

In this section we prove the following theorem:

Theorem 3.3 *Given a triangulation G, there exists a spanning subgraph S of G such that S is a triangulated binary cactus.*

Consider any triangulation G. We are going to construct a binary cactus S spanning G. First, we outline the algorithm to construct S. Such an algorithm has several steps. At the first step, we choose a vertex u incident to the outer face of G and we construct a triangulated cycle C_T composed of u and of all its neighbors. We remove u and its incident edges from G, obtaining a biconnected internally-triangulated plane graph G^*. At the beginning of each step after the first one, we suppose to have already constructed a binary cactus S whose vertices are a subset of the vertices of G (at the beginning of the second step, S coincides with C_T), and we assume to have a set $\mathcal{G}$ of subgraphs of G (at the beginning of the second step, G^* is the only graph in $\mathcal{G}$). Each of such subgraphs is biconnected, internally-triangulated, has an outer face whose vertices already belong to S, and has internal vertices. All such internal vertices do not belong to S and each vertex of G not belonging to S is internal to a graph in $\mathcal{G}$. Only one of the graphs in $\mathcal{G}$ may have chords. During each step, we perform the following two actions:

- *Action 1.* We partition the only graph G_C of $\mathcal{G}$ with chords, if any, into several biconnected internally-triangulated chordless plane graphs; we remove G_C from $\mathcal{G}$ and we add to $\mathcal{G}$ all graphs with internal vertices into which G_C has been partitioned.
- *Action 2.* We choose a graph G_i from $\mathcal{G}$, we choose a vertex u incident to the outer face of G_i and already belonging to exactly one block of S, and we add to S a block composed of u and of all its neighbors internal to

G_i. We remove u and its incident edges from G_i, obtaining a biconnected internally-triangulated plane graph G_i^*. We remove G_i from $\mathcal{G}$ and we add G_i^* to $\mathcal{G}$.

The algorithm stops when $\mathcal{G}$ is empty, that is, when all the vertices of G have been spanned by S. An example of application of the algorithm is shown in the Appendix.

Now we give the details of the above outlined algorithm. At the first step of the algorithm (see Fig. 3.9), choose any vertex u incident to the outer face of G. Consider all the neighbors $(u_1, u_2, \ldots, u_l)$ of u in clockwise order around it. Since G is a triangulation, $C = (u, u_1, u_2, \ldots, u_l)$ is a cycle. Let C_T be the triangulated cycle obtained by adding to C the edges connecting u to its neighbors. Let $S = C_T$. Remove vertex u and all its incident edges from G, obtaining a biconnected internally-triangulated graph G^*.

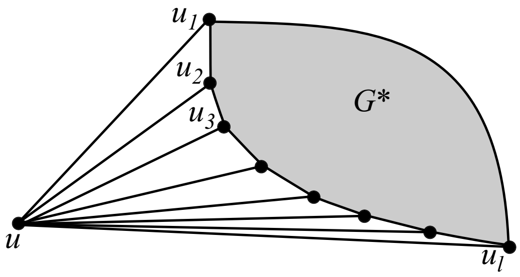


Figure 3.9: First step of the algorithm.

If G^* has no internal vertex, then all the vertices of G belong to S and we have the desired binary cactus spanning G. Otherwise, G^* has internal vertices. Let $\mathcal{G} = \{G^*\}$.

At each step of the algorithm, for each graph $G_i \in \mathcal{G}$, consider the vertices incident to $f(G_i)$. Each of such vertices can be either *forbidden for G_i* or *assigned to G_i*. A vertex w is forbidden for G_i if the choice of not introducing in S any new block incident to w and spanning a subgraph of G_i has been done. Conversely, a vertex w is assigned to G_i if a new block incident to w and spanning a subgraph of G_i could be introduced in S. For example, w is forbidden for G_i if two blocks of S already exist sharing w as a cutvertex. At the end of the first step of the algorithm, choose any two vertices incident to $f(G^*)$ as the only forbidden vertices for G^*. All the other vertices incident to $f(G^*)$ are assigned to G^*.

At the beginning of the i-th step, with $i \geq 2$, we assume that each of the following holds:

- *Invariant A:* Graph S is a binary cactus spanning all and only the vertices that are not internal to any graph in $\mathcal{G}$.
- *Invariant B:* Each graph in $\mathcal{G}$ is biconnected, internally-triangulated, and has internal vertices.
- *Invariant C:* Only one of the graphs in $\mathcal{G}$ may have chords.
- *Invariant D:* No internal vertex of a graph $G_i \in \mathcal{G}$ belongs to a graph $G_j \in G$, with $i \neq j$.
- *Invariant E:* For each graph $G_i \in \mathcal{G}$, all the vertices incident to $f(G_i)$ are assigned to G_i, except for two vertices, which are forbidden.
- *Invariant F:* Each vertex v incident to the outer face of a graph in $\mathcal{G}$ is assigned to at most one graph $G_i \in \mathcal{G}$. The same vertex is forbidden for all graphs $G_j \in \mathcal{G}$ such that v is incident to $f(G_j)$ and $i \neq j$.
- *Invariant G:* Each vertex assigned to a graph in $\mathcal{G}$ belongs to exactly one block of S.

Such invariants clearly hold after the first step of the algorithm. During each step of the algorithm after the first one, we perform the following two actions.

Action 1: If $\mathcal{G}$ does not contain any graph with chords, go to Action 2. Otherwise, by Invariant C, only one of the graphs in $\mathcal{G}$, say G_C, may have chords. We use the chords of G_C to partition it into k biconnected, internally-triangulated, chordless graphs G^j_C, with $j = 1, 2, \ldots, k$.

Consider the subgraph O_C of G_C induced by the vertices incident to $f(G_C)$. Clearly, O_C is a biconnected outerplane graph. To each internal face f of O_C delimited by a cycle c, a graph G^j_C is associated such that G^j_C is the subgraph of G_C induced by the vertices of c or inside c. We are going to replace G_C with graphs G^j_C in $\mathcal{G}$. However, we first show how to decide which vertices incident to the outer face of a graph G^j_C are assigned to G^j_C and which vertices are forbidden for G^j_C. Since each graph G^j_C is univocally associated with a face of O_C, namely the face of O_C delimited by the cycle that delimits $f(G^j_C)$, in the following we assign vertices to the faces of O_C and we forbid vertices for the faces of O_C, meaning that if a vertex is assigned to (forbidden for) a face of O_C delimited by a cycle c then it is assigned to (resp. forbidden for) graph G^j_C whose outer face is delimited by c.

We want to assign the vertices incident to $f(O_C)$ to faces of O_C so that:

- Property 1: No forbidden vertex is assigned to any face of O_C;
- Property 2: No vertex is assigned to more than one face of O_C;
- Property 3: Each face of O_C has exactly two incident vertices which are forbidden for it; all the other vertices of the face are assigned to it.

By Invariant E, G_C has two forbidden vertices. We construct an assignment of vertices to the faces of O_C in some steps. Let p be the number of chords of O_C. Consider the Hamiltonian cycle O^0_C of O_C, and assign all the vertices of O^0_C, but for the two forbidden vertices, to the only internal face of O^0_C. At the i-th step, $1 \leq i \leq p$, we insert into O^{i-1}_C a chord of O_C, obtaining a graph O^i_C. This is done so that Properties 1–3 are satisfied by O^i_C (with O^i_C instead of O_C). After all p chords of O_C have been inserted, $O^p_C = O_C$, and we have an assignment of vertices to faces of O_C satisfying Properties 1–3.

Properties 1–3 are clearly satisfied by the assignment of vertices to the faces of O^0_C. Inductively assume Properties 1–3 are satisfied by the assignment of vertices to the faces of O^{i-1}_C. Let (u_a, u_b) be the chord that is inserted at the i-th step. Chord (u_a, u_b) partitions a face f of O^{i-1}_C into two faces f_1 and f_2. By Property 3, two vertices u^*_1 and u^*_2 incident to f are forbidden for it and all other vertices incident to f are assigned to it. For each face of O^i_C different from f_1 and f_2, assign and forbid vertices as in the same face in O^{i-1}_C. Assign and forbid vertices for f_1 and f_2 as follows:

- If vertices u_a and u_b are the same vertices of u^*_1 and u^*_2 (see Fig. 3.10), assign to each of f_1 and f_2 all the vertices incident to it, except for u_a and u_b. No forbidden vertex has been assigned to any face of O^i_C (Property 1). Vertices u_a and u_b have not been assigned to any face. All the vertices assigned to f belong to exactly one of f_1 and f_2 and so they have been assigned to exactly one face (Property 2). The only vertices of f_1 (resp. of f_2) not assigned to it are u_a and u_b, while all the other vertices are assigned to such a face (Property 3).
- If vertices u_a and u_b are both distinct from each of u^*_1 and u^*_2 and both u^*_1 and u^*_2 are in the same of f_1 and f_2, say in f_1 (see Fig. 3.11), assign to f_1 all the vertices incident to it, except for u^*_1 and u^*_2, and assign to f_2

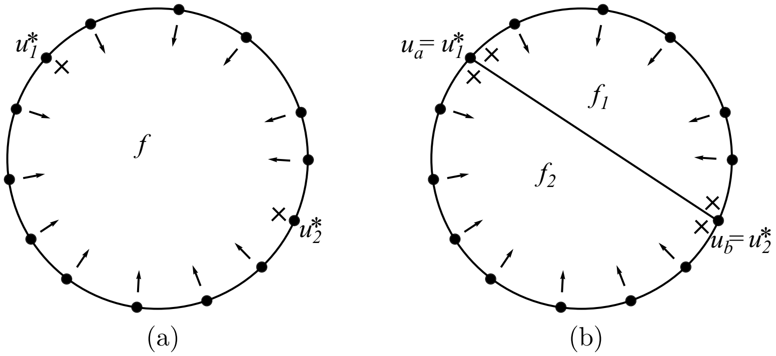


Figure 3.10: Vertices u_a and u_b are the same vertices of u_1^* and u_2^*.

all the vertices incident to it, except for u_a and u_b. No forbidden vertex has been assigned to any face of O_C^i (Property 1). Vertices u_a and u_b have been assigned to exactly one face, namely f_1. All the other vertices assigned to f belong to exactly one of f_1 and f_2 and so they have been assigned to exactly one face (Property 2). The only vertices of f_1 (resp. of f_2) not assigned to it are u_1^* and u_2^* (resp. u_a and u_b), while all the other vertices are assigned to such a face (Property 3).

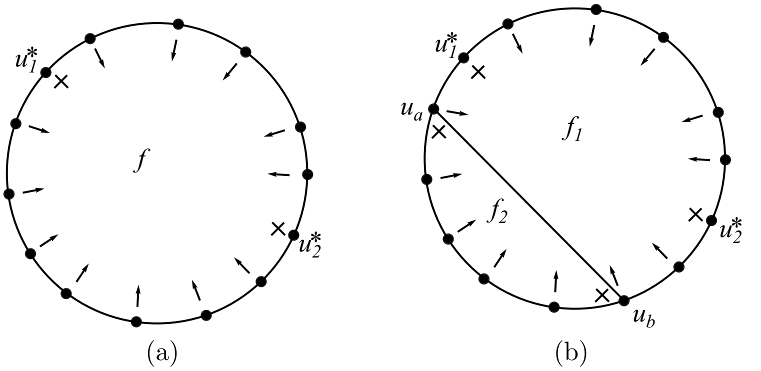


Figure 3.11: Vertices u_a and u_b are both distinct from each of u_1^* and u_2^* and both u_1^* and u_2^* are in f_1.

- If vertices u_a and u_b are both distinct from each of u_1^* and u_2^* and one of u_1^* and u_2^*, say u_1^*, is in f_1 while the other one, say u_2^*, is in f_2 (see Fig. 3.12), assign to f_1 all the vertices incident to it, except for u_1^* and u_a, and assign to f_2 all the vertices incident to it, except for u_2^* and u_b. No forbidden vertex has been assigned to any face of O_C^i (Property 1). Vertices u_a and u_b have been assigned to exactly one face, namely f_2 and f_1, respectively. All the other vertices assigned to f belong to exactly one of f_1 and f_2 and so they have been assigned to exactly one face (Property 2). The only vertices of f_1 (resp. of f_2) not assigned to it are u_1^* and u_a (resp. u_2^* and u_b), while all the other vertices are assigned to such a face (Property 3).

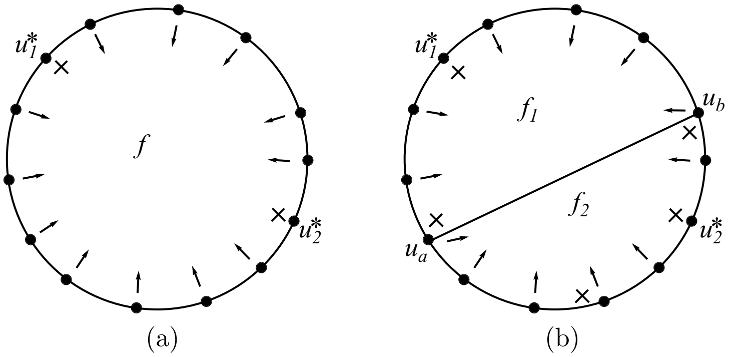
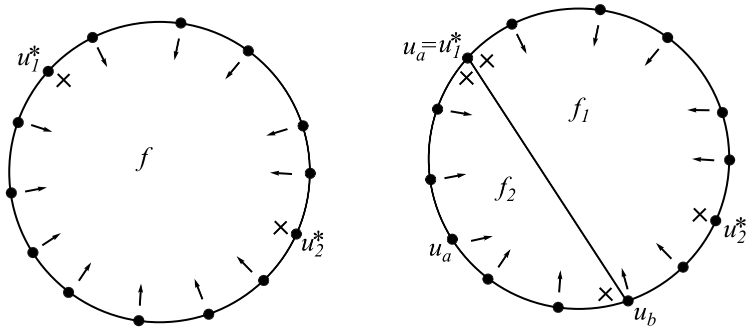


Figure 3.12: Vertices u_a and u_b are both distinct from each of u_1^* and u_2^*, u_1^* is in f_1, and u_2^* is in f_2.

- If one of the vertices u_1^* and u_2^* coincides with one of u_a and u_b, say u_1^* coincides with u_a, and the other one, say u_2^*, is in one of f_1 and f_2, say in f_1 (see Fig. 3.13), assign to f_1 all the vertices incident to it, except for u_2^* and u_a, and assign to f_2 all the vertices incident to it, except for u_a and u_b. No forbidden vertex has been assigned to any face of O_C^i (Property 1). Vertex u_a has not been assigned to any face and vertex u_b has been assigned to exactly one face, namely f_1. All the other vertices assigned to f belong to exactly one of f_1 and f_2 and so they have been assigned to exactly one face (Property 2). The only vertices of f_1 (resp. of f_2) not assigned to it are u_2^* and u_a (resp. u_a and u_b), while all the other vertices are assigned to such a face (Property 3).



(a) (b)

Figure 3.13: Vertex u_1^* coincides with u_a and vertex u_2^* is in f_1.

Graph G_C is removed from $\mathcal{G}$. All the graphs G_C^j having internal vertices are added to $\mathcal{G}$. We prove that Invariants A–G are satisfied after Action 1.

Invariant A: A vertex is internal to a graph in $\mathcal{G}$ after Action 1 if and only if it is internal to a graph in $\mathcal{G}$ before Action 1. Since no block is added to S during Action 1, then Invariant A holds after Action 1.

Invariant B: By construction, each graph G_C^j inserted into $\mathcal{G}$ after Action 1 has internal vertices. Further, G_C^j is the graph contained inside a simple cycle of a biconnected internally triangulated plane graph, hence it is biconnected and internally triangulated, as well, satisfying Invariant B.

Invariant C: By Invariant C, before Action 1 only graph G_C may have chords among the graphs in $\mathcal{G}$. After Action 1, however, G_C is replaced in $\mathcal{G}$ by chordless graphs and hence no graph in $\mathcal{G}$ has chords, satisfying Invariant C.

Invariant D: By Invariant D, each vertex that, before Action 1, is internal to a graph $G_i \neq G_C$ in $\mathcal{G}$ does not belong to any graph $G_j \neq G_i$ in $\mathcal{G}$. Since the set of vertices belonging to graphs G_C^j is a subset of the vertices of G_C, after Action 1 Invariant D holds for all vertices internal to a graph $G_i \neq G_C^j$. An internal vertex of a graph G_C^j is an internal vertex of G_C, as well, hence, by Invariant D, it does not belong to any graph that has not been introduced in $\mathcal{G}$ during Action 1. It remains to prove that an internal vertex of a graph G_C^j does not belong to any graph G_C^l, with

$l \neq j$. By construction, the internal vertices of such graphs are inside cycles corresponding to distinct faces of O_C. Hence, an internal vertex of G_C^j does not belong to G_C^l.

Invariant E: Invariant E holds for all graphs that are in $\mathcal{G}$ before Action 1 and that are still in $\mathcal{G}$ after Action 1. By Property 3, each graph G_C^j inserted into $\mathcal{G}$ after Action 1 satisfies Invariant E.

Invariant F: All vertices that, before Action 1, are assigned to a graph $G_i \neq G_C$ in $\mathcal{G}$ satisfy Invariant F after Action 1. Namely, by Invariant F before Action 1, if they are incident to $f(G_C)$, then they are forbidden for G_C and, by Property 1, they are not assigned to any graph G_C^j. By Invariant F, before Action 1 each vertex w assigned to G_C is not assigned to any graph $G_i \neq G_C$ in $\mathcal{G}$. After Action 1, G_C is not a graph in $\mathcal{G}$ any longer, hence w is not assigned to it. By Property 2, after Action 1 each vertex is assigned to at most one graph G_C^j, hence Invariant F holds after Action 1.

Invariant G: Since no block is added to S during Action 1, and since the set of vertices assigned to graphs in $\mathcal{G}$ after Action 1 is a subset of the set of vertices assigned to graphs in $\mathcal{G}$ before Action 1, then Invariant G holds after Action 1.

Action 2: After Action 1 all graphs in $\mathcal{G}$ are chordless. Notice that there is at least one graph G_i in $\mathcal{G}$, otherwise the algorithm would have stopped before Action 1. By Invariant B, G_i has internal vertices. Choose any vertex u that is incident to $f(G_i)$ and that is assigned to G_i (see Fig. 3.14). By the biconnectivity of G_i and by the fact that it has internal vertices, $f(G_i)$ has at least three vertices. Since each graph in $\mathcal{G}$ has at most two forbidden vertices (by Invariant E), a vertex u assigned to G_i always exists. Consider all the neighbors $(u_1, u_2, \ldots, u_l)$ of u internal to G_i, in clockwise order around u. Since G is biconnected, chordless, internally triangulated, and has internal vertices, then $l \geq 1$. If $l = 1$ then let C_T be edge (u, u_1). Otherwise, let C_T be the triangulated cycle obtained by adding to cycle $(u, u_1, u_2, \ldots, u_l)$ the edges connecting u to its neighbors. Add C_T to S. Remove u and its incident edges from G_i, obtaining a graph G_i^*. Assign to G_i^* all the vertices incident to $f(G_i^*)$, except for the two vertices that are forbidden for G_i. Remove G_i from $\mathcal{G}$ and insert G_i^*, if it has internal vertices, into $\mathcal{G}$.

We prove that Invariants A–G are satisfied after Action 2.

Invariant A: The block $(u, u_1, u_2, \ldots, u_l)$ added to S is either an edge or a triangulated cycle. By Invariant A, before Action 2 all vertices internal

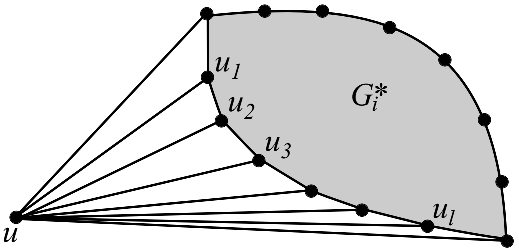


Figure 3.14: Action 2 of a step of the algorithm.

to a graph in $\mathcal{G}$ are not spanned by S. Further, by Invariant G, before Action 2 vertex u belongs to exactly one block of S. It follows that S is still a binary cactus after Action 2. Before Action 2, S spans all and only the vertices that are not internal to graphs in $\mathcal{G}$. The only vertices that are internal to a graph in $\mathcal{G}$ before Action 2 and that are incident to the outer face of a graph in $\mathcal{G}$ after Action 2, are $u_1, u_2, \dots, u_l$, which are spanned by S after Action 2. Hence, S spans all vertices of G that are not internal to a graph in $\mathcal{G}$. Before Action 2, no internal vertex of a graph in G is spanned by S. The vertices which are added to S during Action 2 are incident to $f(G_i^*)$, hence, by Invariant D to be proved below, they are not internal to any graph in $\mathcal{G}$ after Action 2. Hence, S does not span vertices of G that are internal to a graph in $\mathcal{G}$ satisfying Invariant A.

Invariant B: By construction, G_i^* is the only graph inserted into $\mathcal{G}$ after Action 2. However, G_i^* is biconnected and internally triangulated, since it is obtained from a graph G_i that, by Invariant B before Action 2, is biconnected, internally triangulated, chordless, and has internal vertices, by removing a vertex incident to $f(G_i)$. Further, G_i^* has internal vertices, otherwise it would not have been inserted into $\mathcal{G}$. Hence, Invariant B is satisfied after Action 2.

Invariant C: Before Action 2, all graphs in $\mathcal{G}$ have no chord. At most one graph, G_i^*, is inserted into $\mathcal{G}$ after Action 2, hence Invariant C is still satisfied.

Invariant D: By Invariant D, before Action 2 no internal vertex of a graph $G_l \neq G_i$ in $\mathcal{G}$ belongs to a graph $G_j \neq G_l$ in $\mathcal{G}$. Since the vertices of G_i^*

are a subset of the vertices of G_i then, after Action 2, Invariant D holds for each internal vertex of G_l. Further, the internal vertices of G_i^* are a subset of the internal vertices of G_i and hence, after Action 2, Invariant D holds also for each internal vertex of G_i^*.

Invariant E: Invariant E holds for all the graphs that are in $\mathcal{G}$ before Action 2 and that are still in $\mathcal{G}$ after Action 2. By construction, all the vertices incident to the outer face of G_i^*, except for the two forbidden vertices of G_i, are assigned to G_i^*, satisfying Invariant E.

Invariant F: The only vertices that are assigned to a graph in $\mathcal{G}$ during Action 2 are the vertices incident to the outer face of G_i^*. All the vertices internal to G_i before Action 2 and incident to the outer face of G_i^* after Action 2 are assigned only to G_i^*, namely if before Action 2 one of such vertices is assigned to a graph $G_j \neq G_i$, then such a vertex would be incident to the outer face of G_j, contradicting Invariant D. All the vertices that are assigned to G_i before Action 2 and that are incident to the outer face of G_i^* after Action 2, are assigned only to G_i before Action 2, by Invariant F, and hence they are assigned only to G_i^* after Action 2. All the vertices that are assigned to a graph different from G_i are such that, if they are incident to the outer face of G_i, then they are forbidden for it. Since all the vertices forbidden for G_i are forbidden for G_i^*, then Invariant F holds for such vertices, as well.

Invariant G: The block added to S after Action 2 spans only vertices internal to G_i and vertex u. Hence, all the vertices assigned to a graph in $\mathcal{G}$ and not belonging to G_i are still spanned by a single block of S. All the vertices incident to the outer face of G_i, except for u, are not spanned by the block added during Action 2. All the vertices internal to G_i and assigned to G_i^* are spanned by the only block added during Action 2. Finally, after Action 2, vertex u is not assigned to any graph in $\mathcal{G}$ any longer.

When the algorithm stops, i.e., when there is no graph in $\mathcal{G}$, by Invariant A graph S is a binary cactus spanning all vertices of G, hence proving Theorem 3.3.

3.5 Extension to Triconnected Planar Graphs

In this section, we show how slight modifications of the two main arguments (see Sect. 3.3 and Sect. 3.4) used to prove that every triangulation has a greedy drawing allow to prove Conjecture 3.1. First, we show how to construct a greedy drawing of any *non-triangulated binary cactus*, that is a connected graph such that: (i) the block associated with each B-node of $\mathcal{T}$ is either an edge or a simple cycle; and (ii) every cutvertex is shared by exactly two blocks of S. Second, we show that a triconnected planar graph can always be spanned by a non-triangulated binary cactus. Notice that a non-triangulated binary cactus is easily obtained from a triangulated binary cactus by removing the edges internal to the triangulated cycles.

It is not difficult to argue that the algorithm shown in Sect. 3.3 also constructs greedy drawings of any non-triangulated binary cactus S. More specifically, construct the BC-tree $\mathcal{T}$ of S; consider each block $(r(\mu) = u_0, u_1, \dots, u_{h-1})$ corresponding to a B-node μ of $\mathcal{T}$ and insert a dummy edge between $r(\mu)$ and each node u_i, with $1 \leq i \leq h-2$; the resulting graph S' is a triangulated binary cactus; apply the algorithm described in Sect. 3.3 to construct a greedy drawing Γ' of S'; finally, remove dummy edges from Γ', obtaining a drawing Γ of S.

We claim that Γ is a greedy drawing. Notice that the validity of Lemmata 3.1 and 3.2 only depends on the angles of the geometric construction. Hence, such Lemmata hold for Γ. Then, it is enough to prove that at each step of the induction Γ satisfies Properties 1–4 described in Sect. 3.3.

Actually, Property 2 and Property 4 are trivially verified, since they only depend on the angles of the construction.

The proof of Property 1 can be conduced analogously to the one presented in Sect. 3.3, namely by proving that, for every pair of vertices w_1 and w_2, there exists a distance-decreasing path between them. However, the case in which the distance-decreasing path contains edge $(u_i, r(\mu))$, for some $2 \leq i \leq h-2$, deserves an explicit discussion, because such an edge is no longer an edge of the graph. Observe that it can be supposed that one out of w_1 and w_2 is $r(\mu)$, because in all the other cases the distance-decreasing path between w_1 and w_2 does not contain $(u_i, r(\mu))$.

First, suppose that the path ends at $r(\mu)$, i.e., $w_2 = r(\mu)$. Edge $(u_i, r(\mu))$ can be replaced either by path $(u_i, u_{i-1}, \dots, u_1, u_0)$, if $i \leq h/2$, or by path $(u_i, u_{i+1}, \dots, u_{h-1}, u_0)$, if $i \geq h/2$, still leaving the path distance-decreasing. In fact (see Fig. 3.15 (a)), denote by p' the intersection point between C' and segment $\overline{cp^*}$ and suppose that $i \geq h/2$, the case in which $i \leq h/2$ being analogous; angle $\widehat{u_iu_{i+1}p'}$ is greater than or equal to $\pi/2$ because triangle (u_i, u_{i+1}, p') is inscribed in no more than half a circumference with u_{i+1} as middle point; then, angle $\widehat{u_iu_{i+1}p^*}$ is also greater than $\pi/2$ because it is strictly greater than $\widehat{u_iu_{i+1}p'}$; hence, $\overline{p^*u_i}$ is longer than $\overline{p^*u_{i+1}}$; it follows that, when traversing edge (u_i, u_{i+1}), the path decreases its distance from the point p^* where $r(\mu)$ is drawn.

Second, suppose that the path starts at $r(\mu)$, i.e., $w_1 = r(\mu)$. Edge $(u_i, r(\mu))$ can be replaced either by path $(u_i, u_{i-1}, \dots, u_1, u_0)$, if $i \leq h/2$, or by path $(u_i, u_{i+1}, \dots, u_{h-1}, u_0)$, if $i \geq h/2$, still leaving the path distance-decreasing. In fact, suppose that $i \geq h/2$, the case in which $i \leq h/2$ being analogous; as in the previous case, edge $(r(\mu), u_{h-1})$ can be shown to decrease the distance from w_2 by considering triangle $(r(\mu), u_{h-1}, w_2)$ and arguing that angle $p^*\widehat{u_{h-1}w_2}$ is greater than $\pi/2$. Further, path $(u_{h-1}, u_{h-2}, \dots, u_{i+1}, u_i, \dots, w_2)$ can be shown to be distance-decreasing as in the proof of Property 1 in Sect. 3.3 (in the case in which w_1 belongs to $S(\mu_i)$ and w_2 belongs to $S(\mu_j)$).

In order to prove Property 3, it is sufficient to observe that an edge $(u_i, r(\mu))$ can be replaced either by path $(u_i, u_{i-1}, \dots, u_1, u_0)$ or by path $(u_i, u_{i+1}, \dots, u_{h-1}, u_0)$, still obtaining a path in which at every step the distance from any point in $W(p^*)$ decreases. In fact (see Fig.3.15 (b)), denote by p any point inside $W(p^*)$, and denote by $a_{i-1,i}$ and $a_{i,i+1}$ the axes of segments $\overline{p_{i-1}p_i}$ and $\overline{p_ip_{i+1}}$, respectively. Since $a_{i-1,i}$ and $a_{i,i+1}$ intersect in the center of C', we have that p is either to the left of $a_{i-1,i}$ or to the right of $a_{i,i+1}$, or both. Suppose that p is to the right of $a_{i,i+1}$, the other case being analogous. Then, $d(p, p_{i+1}) < d(p, p_i)$. The repetition of such an argument leads to prove that path $(u_i, u_{i+1}, \dots, u_{h-1}, u_0)$ decreases the distance from p at every vertex.

Since there exists an algorithm to construct greedy drawings of non-triangulated binary cactuses, in order to prove Conjecture 3.1 it suffices to show that every triconnected planar graph admits a non-triangulated binary cactus as a spanning subgraph. In the following we sketch how to extend the arguments of Sect. 3.4 in order to prove such a result.

The algorithm to find a non-triangulated binary cactus spanning a given triconnected planar graph G consists of several steps, in which the cactus is constructed incrementally by adding to it one block at a time. As in the triangulated case, at the beginning of each step after the first one, we suppose to have already constructed a non-triangulated binary cactus S whose vertices are a subset of the vertices of G, and we assume to have a set $\mathcal{G}$ of subgraphs of G. Further, we assume that the following invariants hold:

- *Invariant A:* Graph S is a non-triangulated binary cactus spanning all

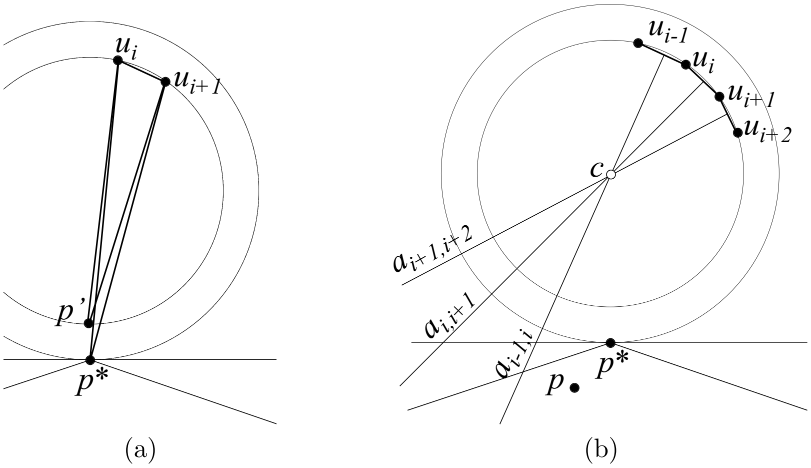


Figure 3.15: (a) When traversing edge (u_i, u_{i+1}), the distance from p^* decreases. (b) When traversing edge (u_i, u_{i+1}), the distance from p decreases.

and only the vertices that are not internal to any graph in $\mathcal{G}$.

- *Invariant B:* Each graph in $\mathcal{G}$ is biconnected and has internal vertices.

- *Invariant C:* At most one graph $G_C \in \mathcal{G}$ has separating pairs. However, if G_C exists, each of its separating pairs has both vertices incident to $f(G_C)$.

- *Invariant D:* No internal vertex of a graph $G_i \in \mathcal{G}$ belongs to a graph $G_j \in G$, with $i \neq j$.

- *Invariant E:* For each graph $G_i \in \mathcal{G}$, all the vertices incident to $f(G_i)$ are assigned to G_i, except for two vertices, which are forbidden.

- *Invariant F:* Each vertex v incident to the outer face of a graph in $\mathcal{G}$ is assigned to at most one graph $G_i \in \mathcal{G}$. The same vertex is forbidden for all graphs $G_j \in \mathcal{G}$ such that v is incident to $f(G_j)$ and $i \neq j$.

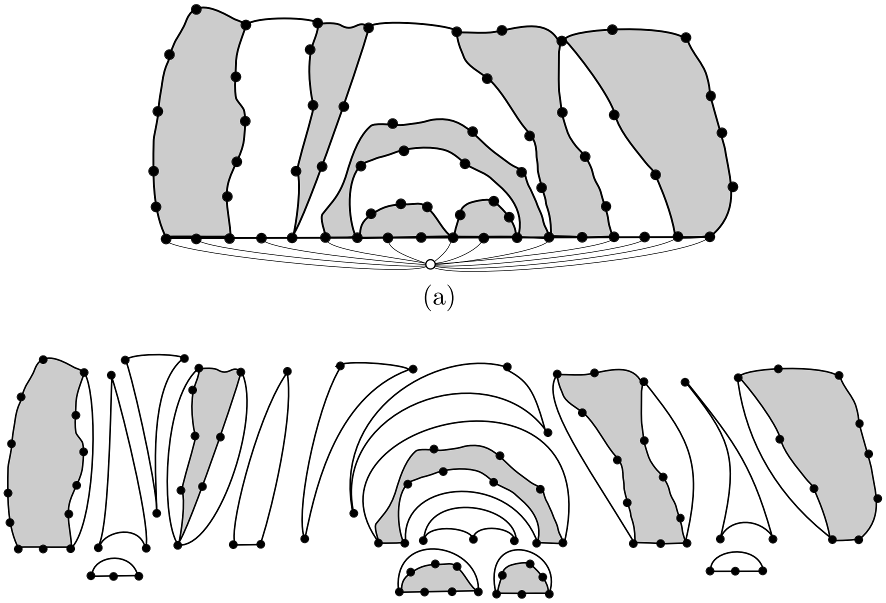
- *Invariant G:* Each vertex assigned to a graph in $\mathcal{G}$ belongs to exactly one block of S.

During each step, we perform two different actions. Action 1 removes from $\mathcal{G}$ the only graph G_C which contains separating pairs, if such a graph exists, and partitions G_C into a set of triconnected planar graphs G_C^i to be added to $\mathcal{G}$. Action 2 removes from a graph $G_i \in \mathcal{G}$ a vertex incident to $f(G_i)$ and its incident edges and creates a new block to be added to S. At the end of each of the two actions, Invariants A–G are satisfied. The algorithm stops when $\mathcal{G}$ is empty, that is, when all the vertices of G have been spanned by S.

A first difference between the triangulated and the non-triangulated case concerns the first step of the algorithm. Namely, while in the triangulated case we select one vertex v of the outer face and we initialize the cactus with the block composed of v and of its neighbors, in this new algorithm we initialize the cactus with the cycle delimiting the outer face.

Another important difference is in Action 1, that is, in the way the graph G_C which may be not triconnected is partitioned into subgraphs. In the triangulated case, such a partition is done by considering the chords of $f(G_C)$. In the non-triangulated case we have to more generally consider separating pairs incident to $f(G_C)$, since we are not guaranteed that every two vertices composing a separating pair are joined by an edge. Refer to Fig. 3.16. The partition is performed by considering one separating pair at a time. At the beginning of every step of such an algorithm, we have a partition of G_C into a set of graphs G_i. Each graph G_i which still has a separating pair is further partitioned into two subgraphs G_i^1 and G_i^2 and each of the vertices incident to $f(G_i)$ is assigned to, or forbidden for, G_i^1 and G_i^2 by means of the same algorithm described in Sect. 3.4. Hence, the assignment of the vertices to the graphs G_i^1 and G_i^2 can be done so that the invariant that each of G_i^1 and G_i^2 has at most two forbidden vertices is maintained. A dummy edge connecting the two vertices of the separating pair has to be added incident to the outer face of each of G_i^1 and G_i^2, if it does not exist yet, in order to maintain the invariant that all the vertices incident to the outer faces of G_i^1 and G_i^2 have already been assigned to some block of S. Such a dummy edge is incident to the outer faces of G_i^1 and G_i^2 and hence it will not be part of any new block that is added to S in the following steps of the algorithm. It is easy to see that the described procedure for partitioning a graph into subgraphs does not introduce new separating pairs, does not introduce multiple edges, and hence it terminates providing a set of triconnected plane graphs.

Concerning Action 2, while in the triangulated case at every step we add to the cactus either an edge or a triangulated cycle, in the non-triangulated case we add either an edge or a simple cycle. Such a cycle is obtained as follows (see Fig. 3.17). As in the triangulated case, consider a vertex v incident to the



(a)

(b)

Figure 3.16: Partition of a biconnected graph having all separating pairs incident to the outer face into a set of triconnected planar graphs.

outer face of a subgraph $G_i \in \mathcal{G}$ and such that v is assigned to G_i. Consider the internal faces of G_i that are incident to v, except for the two faces f_1 and f_2 sharing an edge with $f(G_i)$. Add to S the cycle that passes through all the vertices that are incident to such faces. Remove vertex v and its incident edges from G_i, obtaining a new graph G_i^*. Consider the two vertices v_1' and v_1'' adjacent to v and belonging to f_1. A dummy edge (v_1', v_1'') is added to G_1^*, if it does not exist yet, incident to $f(G_i^*)$. Analogously, consider the two vertices v_2' and v_2'' adjacent to v and belonging to f_2 and add a dummy edge (v_2', v_2'') to G_i^*, if it does not exist yet, incident to $f(G_i^*)$. Such dummy edges allow to maintain the invariant that all the vertices incident to $f(G_i^*)$ have already been assigned to some block of S.

We choose to present the algorithm for triangulations as the main contribution of this chapter because a proof of Conjecture 3.1 was very recently and

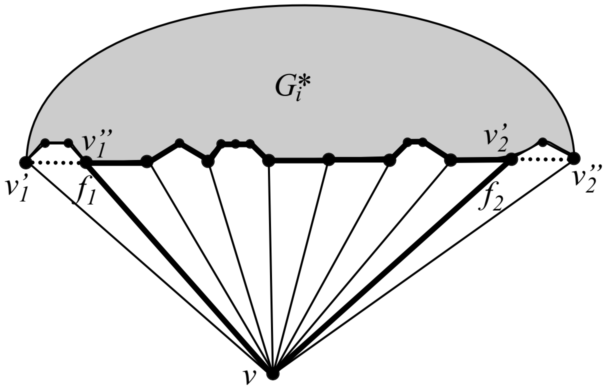


Figure 3.17: Action 2. The thick cycle is added to S. The dotted edges are inserted incident to $f(G_i^*)$, in order to maintain the invariant that all the vertices incident to $f(G_i^*)$ have already been assigned to some block of S.

independently presented by Leighton and Moitra at FOCS'08 [LM08]. Surprisingly, the approach used by Leighton and Moitra is exactly the same as ours. In fact, in [LM08] the authors define a class of graphs, called *Christmas cactus graphs*, which coincides with the class of non-triangulated binary cactuses; they show an algorithm to construct greedy drawings of Christmas cactus graphs and they show that every triconnected planar graph is spanned by a Christmas cactus graph. However, the way such results are achieved differs from ours. Such an issue is discussed below.

Concerning the geometric construction of greedy drawings of Christmas cactus graphs, the algorithm by Leighton and Moitra is quite similar to ours, even if a slightly different construction is used. Their algorithm places the nodes of the graph on a set of concentric circumferences $C_0, C_1, \ldots, C_k$, so that the block corresponding to the root ν of the BC-tree $\mathcal{T}$ has its nodes placed on C_0 and each block μ at depth i (where the depth is meant to be the number of B-nodes in the path from ν to μ in $\mathcal{T}$) is placed on C_i, except for the C-node parent of μ, which is placed on C_{i-1}. The difference between the radii of two consecutive circumferences (and hence the length of the edges of the drawing) exponentially decreases with i.

Concerning the construction of a Christmas cactus graph spanning a given triconnected planar graph, we have the main differences between our techniques and Leighton and Moitra's ones. In fact, in order to show that every triconnected planar graph is spanned by a Christmas cactus graph, they use some results from a paper [GR94] by Gao and Richter.

Define a *circuit graph* to be an ordered pair (G, C) such that: (1) G is 2-connected and C is a polygon in G; (2) there exists an embedding of G in the plane such that C bounds a face; and (3) every separating pair of G has both vertices belonging to C. Hence, circuit graphs are a superclass of triconnected planar graphs. Define a *chain of blocks* $B_{i,1}, b_{i,1}, B_{i,2}, b_{i,2}, \dots, B_{i,k_i-1}, b_{i,k_i-1}, B_{i,k_i}$ to be a connected graph such that each block contains at most two cutvertices and each cutvertex is shared by exactly two blocks.

In [GR94], Gao and Richter prove some strong structural results about circuit graphs, which are briefly described below. Gao and Richter prove that, given a circuit graph (G, C) and given two vertices x and y belonging to C, there exists a partition of $V(G)-V(C)$ into subsets $V_1, V_2, \dots, V_m$ and there exist distinct vertices $v_1, v_2, \dots, v_m \in V(C)-\{x, y\}$ such that: (i) the subgraph induced by $V_i \cup \{v_i\}$ is a chain of blocks $B_{i,1}, b_{i,1}, B_{i,2}, b_{i,2}, \dots, B_{i,k_i-1}, b_{i,k_i-1}, B_{i,k_i}$, and (ii) $v_i \in V(B_{i,1}) \setminus \{b_{i,1}\}$.

Gao and Richter used this structural result in order to inductively prove that every triconnected planar graph (in fact, every circuit graph) has a closed 2-walk, which is a walk on the graph starting and ending at the same vertex and passing through each vertex of the graph at least once and at most twice.

The same result is used by Leighton and Moitra to inductively prove that, for every circuit graph (G, C) (and hence every triconnected planar graph G), a Christmas cactus graph S spanning G exists. In fact, the outline of their algorithm for spanning G consists of the following steps: (i) use Gao and Richter's structural result to find chains of blocks $B_{i,1}, b_{i,1}, B_{i,2}, b_{i,2}, \dots, B_{i,k_i-1}, b_{i,k_i-1}, B_{i,k_i}$ spanning all vertices of G not in C; (ii) inductively compute Christmas cactus graphs spanning each block $B_{i,j}$ (which is in turn a circuit graph); (iii) glue the Christmas cactus graphs spanning the blocks and C into a unique Christmas cactus graph spanning G.

Our spanning algorithm, as discussed above, finds the spanning graph of G without using Gao and Richter's result. Moreover, once one has a non-triangulated binary cactus spanning a triconnected planar graph G, it is easy to find a closed 2-walk that passes only through the edges of such a spanning graph. Hence, our algorithm for spanning triconnected planar graphs also provides a proof that every triconnected planar graph has a closed 2-walk alternative to Gao and Richter's one.

It is interesting to observe that our algorithm for spanning a triconnected planar graph with a non-triangulated binary cactus works more generally for circuit graphs (as the Leighton and Moitra's algorithm). In fact, in our algorithm, the only graph which may contain separating pairs before Action 1 is actually a circuit graph, since all its separating pairs are incident to the outer face. A spanning cactus for such a graph can hence be found with the same algorithm described above.

3.6 Conclusions and Open Problems

In this chapter we have shown an algorithm for constructing greedy drawings of triangulations. The algorithm relies on two main results. The first one states that every triangulated binary cactus admits a greedy drawing. The second one states that, for every triangulation G, there exists a triangulated binary cactus S spanning G. Then, we have shown how to modify the algorithm provided for triangulations in order to deal with triconnected planar graphs thus proving a conjecture by Papadimitriou and Ratajczak [PR05], that was independently settled by Leighton and Moitra [LM08].

Although greedy drawings have been proved to be feasible for large classes of planar graphs (like triconnected planar graphs), a characterization of the graphs that admit a greedy drawing seems still to be an elusive goal.

Open Problem 3.1 *Characterize the class of (planar) graphs that admit a greedy drawing.*

The main drawback of our algorithm (and of Leighton and Moitra’s algorithm, as well) is that it uses real coordinates, hence it constructs drawings requiring exponential area once a finite resolution rule has been fixed. It would be interesting to understand whether greedy drawings of every triconnected planar graph can be constructed in polynomial area or whether such drawings require exponential area instead.

Open Problem 3.2 *Which are the asymptotic bounds for the area requirements of greedy drawings of triconnected planar graphs?*

Clearly, the above problem seems to be interesting also when the class of graphs considered is, e.g., the one of triangulations, or the one of greedy-drawable trees.

A stronger version of the Papadimitriou and Ratajczak’s conjecture [PR05] says that for every triconnected planar graph there exists a convex greedy drawing. Such a conjecture has so far not been proved nor disproved, as far as we know.

Open Problem 3.3 *Does a convex greedy drawing of every triconnected planar graph exist?*

Appendix: Spanning a Triangulation with a Binary Cactus, an Example of Application

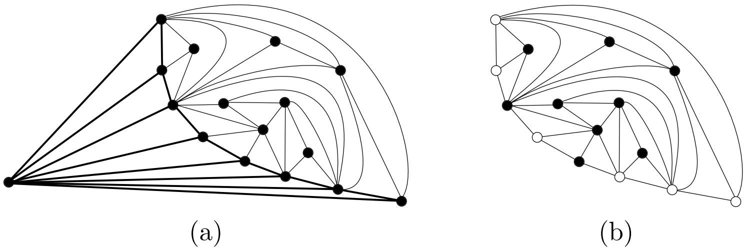


Figure 3.18: First step of the algorithm: (a) A triangulation G, from which a vertex u and its neighbors are selected. The thick subgraph is the triangulated cycle C_T such that $S = C_T$ after Step 1. (b) Graph G^* obtained from G by removing u and its incident edges. Two arbitrarily chosen vertices (represented by black circles) incident to $f(G^*)$ are forbidden for G^*, all others (represented by white circles) are assigned to it.

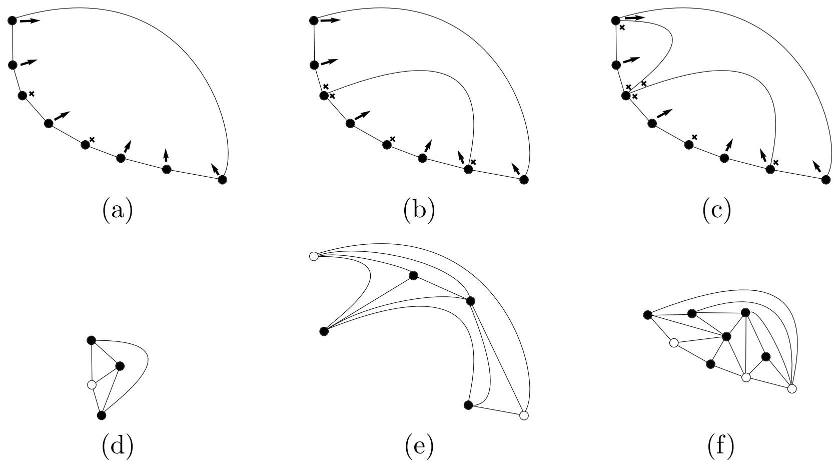


Figure 3.19: Step 2, Action 1. (a)–(c) Outerplane graphs O_C^0, O_C^1, and $O_C^2 = O_C$, and the assignment of vertices to their faces. (d)–(f) Graphs G_1, G_2, and G_3, where $\mathcal{G} = \{G_1, G_2, G_3\}$, obtained by partitioning G^* in biconnected, internally triangulated, chordless subgraphs.

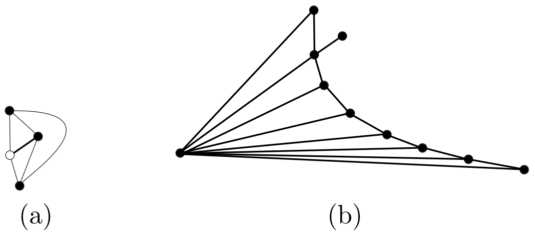


Figure 3.20: Step 2, Action 2. (a) Choice of a graph G_i in $\mathcal{G}$ (here $G_i = G_1$) and of a vertex u incident to $f(G_i)$. The thick subgraph is the edge (u, u_1) added to S after Action 2 of Step 2. (b) Binary cactus S after Action 2 of Step 2. Set $\mathcal{G}$ is now $\{G_2, G_3\}$.

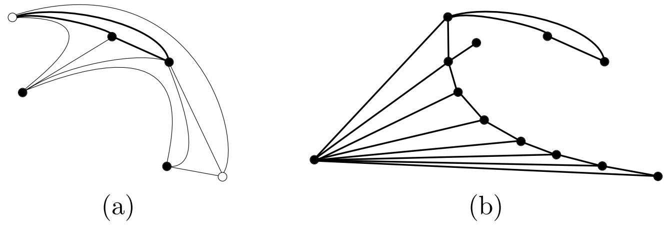


Figure 3.21: Step 3, Action 2 (Action 1 of Step 3 is skipped because no graph in $\mathcal{G}$ has chords). (a) Choice of a graph G_i in $\mathcal{G}$ (here $G_i = G_2$) and of a vertex u incident to $f(G_i)$. The thick subgraph is the triangulated cycle C_T added to S after Step 3, Action 2. (b) Binary cactus after Action 2 of Step 3. Set $\mathcal{G}$ is now $\{G_3\}$.

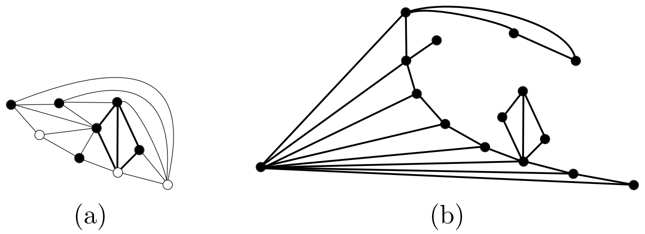


Figure 3.22: Step 4, Action 2 (Action 1 of Step 4 is skipped because no graph in $\mathcal{G}$ has chords). (a) Choice of a graph G_i in $\mathcal{G}$ (here $G_i = G_3$) and of a vertex u incident to $f(G_i)$. The thick subgraph is the triangulated cycle C_T added to S after Step 4, Action 2. (b) Binary cactus S after Action 2 of Step 4. Set $\mathcal{G}$ is now $\{G_3^*\}$, where G_3^* is the graph obtained from G_3 by removing u and its incident edges.

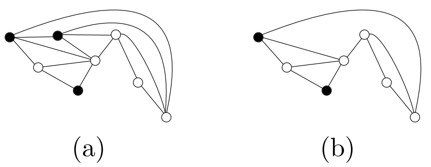


Figure 3.23: Step 5, before Action 1. (a) The only graph $G_C = G_3^*$ in $\mathcal{G}$, with its assigned vertices (white circles) and forbidden vertices (black circles). (b) The outerplane graph O_C induced by the vertices incident to $f(G_C)$.

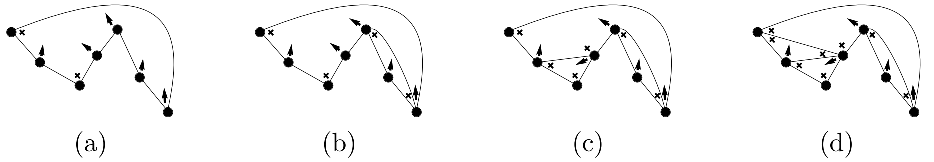


Figure 3.24: Step 5, Action 1. (a)–(d) Outerplane graphs O_C^0, O_C^1, O_C^2, $O_C^3 = O_C$, and the assignment of vertices to their faces. Partitioning G_C into subgraphs G_C^j produces only one graph, say G_4, with internal vertices. Hence, set $\mathcal{G}$ is now $\{G_4\}$.

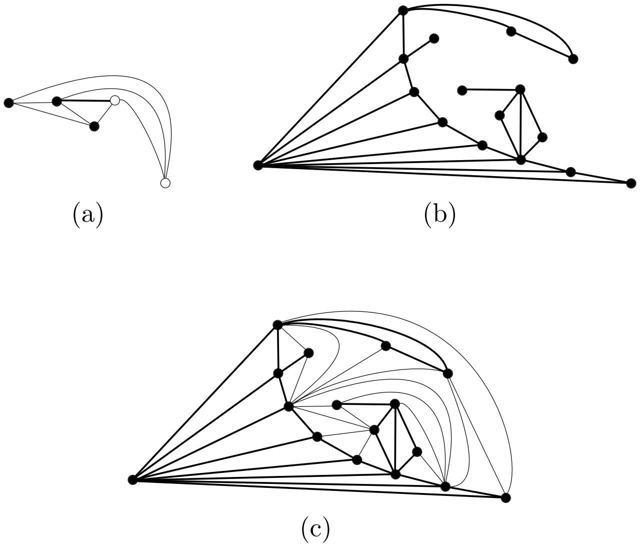


Figure 3.25: Step 5, Action 2: (a) Choice of a graph G_i in $\mathcal{G}$ (here $G_i = G_4$) and of a vertex u incident to $f(G_i)$. The thick subgraph is the edge (u, u_1) added to S after Step 5, Action 2. (b) Binary cactus S at the end of the algorithm. (c) The obtained binary cactus S spans G.

Part II

Series-Parallel Graphs and Outerplanar Graphs

Chapter 4

Straight-line and Poly-line Drawings of Series-Parallel Graphs

In this chapter[1], we consider straight-line and poly-line drawings of series-parallel graphs in small area. We give a sketch of the algorithms achieving the best known area upper bounds at the state of the art and we prove that there exist series-parallel graphs requiring $\Omega(n \log n)$ area in any straight-line or poly-line grid drawing. Such a result is achieved by proving that, in any straight-line or poly-line drawing of $K_{2,n}$, one side of the bounding box has length $\Omega(n)$.

4.1 Introduction

As discussed in Chapter 2, every planar graph admits a planar straight-line (or poly-line) drawing in an $O(n) \times O(n)$ grid, and such a bound is worst-case optimal. Hence, it is natural to search for interesting subclasses of planar graphs admitting subquadratic area drawings. However, several natural subclasses of planar graphs still contain graphs requiring quadratic area in any grid embedding.

Every 4-*connected plane graph* whose outer face has at least four vertices admits a straight-line drawing in $\lfloor \frac{n}{2} \rfloor \times \left(\lceil \frac{n}{2} \rceil - 1 \right)$ area. Such a result, which

[1] The contents of this chapter appeared in [Fra08a]. Thanks to Giuseppe Di Battista for useful discussions.

mainly relies on the construction of two canonical orderings partitioning the graph into two subgraphs that are separately drawn with some visibility properties, was shown by Miura, Nakano, and Nishizeki in [MNN01], improving upon previous results by He [He97]. Such a bound is tight, as shown by the graph in Fig. 4.1 (a).

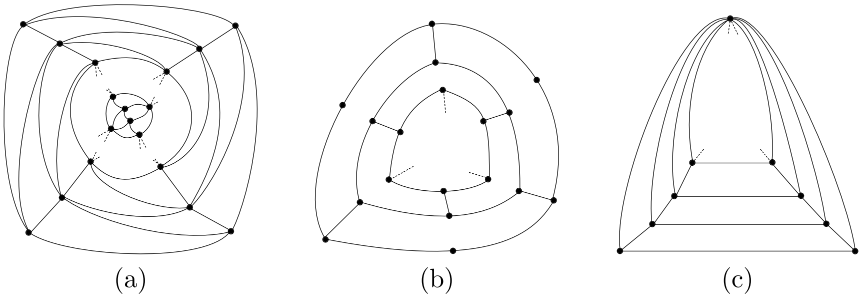


Figure 4.1: (a) A 4-connected plane graph requiring $\lfloor \frac{n}{2} \rfloor \times \left(\lceil \frac{n}{2} \rceil - 1 \right)$ area in any poly-line drawing. (b) A plane graph with degree 3 requiring quadratic area in any poly-line drawing. (c) A plane graph with outerplanarity 2 requiring quadratic area in any poly-line drawing.

Every *bipartite plane graph* admits a straight-line drawing in $\lfloor \frac{n}{2} \rfloor \times \left(\lceil \frac{n}{2} \rceil - 1 \right)$ area. Such a result, that was shown by Biedl and Brandenburg in [BB05], mainly relies on the previously cited result by Miura *et al.* [MNN01]. In fact, Biedl and Brandenburg use the results of Biedl, Kant, and Kaufmann [BKK97] to prove that every planar bipartite graph, except for the *star graph*, can be augmented to a 4-connected plane graph by adding dummy edges to it; this augmentation results in a plane graph whose outer face has three incident vertices; however, an edge incident to the outer face can be removed, obtaining a plane graph which is *almost* 4-*connected* (in [BB05], Biedl and Brandenburg define such a graph as a plane graph that becomes 4-connected when adding an edge incident to the outer face); finally, Biedl and Brandenburg show how Miura *et al.*'s algorithm can be modified in order to work for almost 4-connected plane graphs. The upper bound of Biedl and Brandenburg is tight, since there exist planar bipartite graphs, very similar to the one shown by Miura *et al.*, requiring $\lfloor \frac{n}{2} \rfloor \times \left(\lceil \frac{n}{2} \rceil - 1 \right)$ area in any poly-line drawing.

Graphs with *degree at most three* exist requiring quadratic area in any poly-line/straight-line grid drawing and graphs with *outerplanarity at most two* exist requiring quadratic area in any poly-line grid drawing. Refer to Figs. 4.1 (b) and 4.1 (c) (the graphs shown in Fig. 4.1 (c) was presented by Biedl in [Bie05]).

Planar graphs exclude K_5 and $K_{3,3}$ as minors. Which are the classes of graphs excluding graphs *smaller than* K_5 and $K_{3,3}$ as minors? The answer to the previous question is a list of some of the most studied subclasses of planar graphs. In fact, *trees* are the graphs excluding K_3 as a minor, *outerplanar graphs* are the graphs excluding K_4 and $K_{2,3}$ as minors, and *series-parallel graphs* are the graphs excluding K_4 as a minor. Such graph classes, apart from having nice characterizations in terms of excluded minors, apart from having nice alternative characterizations (as discussed in Chapter 1, trees are connected acyclic graphs, outerplanar graphs admit planar embeddings in which all the vertices are incident to the same face, and series-parallel graphs can be defined inductively from a sequence of series and parallel compositions), and apart of being of real interest for applications, do admit grid drawings in subquadratic area.

The state of the art concerning small-area straight-line and poly-line grid drawings of outerplanar graphs and trees will be discussed in Chapters 5 and 6, respectively. In this chapter, we deal with series-parallel graphs, a class of planar graphs that has been widely investigated in Graph Theory and Graph Drawing (see, e.g., [VTL82, Epp92, BCB$^+$94, Gia03, GDLW06]).

The main algorithmic result about the construction of small-area grid drawings of series-parallel graphs is that every series-parallel graph admits a poly-line drawing in $O(n^{3/2})$ area. Such a bound was proved by Biedl in [Bie05].

In that paper, Biedl first introduces the concept of *flat visibility representation* of a series-parallel graph, that is, a grid drawing in which each vertex is represented by a box with height equal to one and each edge is either a horizontal or a vertical segment. Biedl shows how such representations can be turned into poly-line drawings with asymptotically the same area (in fact, this is done by means of an algorithm reminiscent of the Di Battista and Tamassia's one for turning visibility representations into poly-line drawings, see [DT88] and Chapter 2). Then, Biedl shows how to construct a flat visibility representation of any maximal series-parallel graph in small area.

This is done by means of an algorithm that constructs a drawing of a series-parallel graph G in which the boxes representing the two terminals of G are placed at the top-right corner and at the bottom-right corner of the bounding-box of the drawing. Such an algorithm works by induction on the tree of series/parallel compositions that construct G. At each step, the drawings $\Gamma_{i,1}, \Gamma_{i,2}, \dots, \Gamma_{i,l}$ of graphs $G_{i,1}, G_{i,2}, \dots, G_{i,l}$ are placed together in order to get a drawing of a graph G_i (at the last step of the algorithm, $G_i \equiv G$). The way the drawings of graphs G_i are placed together varies according to several cases, namely, different construction are used if: (i) $G_{i,1}, G_{i,2}, \dots, G_{i,l}$ are in parallel composition; (ii) $G_{i,1}, G_{i,2}$ are in series composition (observe that by the maximality of G, a graph obtained by series composition is composed by exactly two subgraphs), and one of them is an edge; (iii) $G_{i,1}, G_{i,2}$ are in series composition, none of them is an edge, and $G_{i,2}$ is composed by less than $3\sqrt{n_{i,1}} + 1$ subgraphs, where $n_{i,1}$ is the number of vertices of $G_{i,1}$; and (iv) $G_{i,1}, G_{i,2}$ are in series composition, none of them is an edge, and $G_{i,2}$ is composed by more than $3\sqrt{n_{i,1}} + 1$ subgraphs.

Let f be the *fan-out* of a series-parallel graph G, that is, the maximum number of components in a parallel composition for the construction of G. The described approach yields to the following two theorems:

Theorem 4.1 *(Biedl [Bie05]) Any series-parallel graph has a poly-line drawing in $O(n^{3/2})$ area.*

Theorem 4.2 *(Biedl [Bie05]) Any series-parallel graph has a poly-line drawing in $O(fn \log n)$ area.*

While poly-line drawings can be realized in $O(n^{3/2})$ area, no sub-quadratic area upper bound is known in the case of straight-line drawings. In [Bie05], Biedl also proved a $\Omega(\frac{n \log n}{\log \log n})$ area lower bound for straight-line drawings of series-parallel graphs.

The $\Omega(\frac{n \log n}{\log \log n})$ area lower bound for straight-line drawings of series-parallel graphs is a direct consequence of the results in [BCLO03], where Biedl, Chan, and López-Ortiz, settling in the positive a conjecture of Felsner *et al.* [FLW03], proved that no linear-area straight-line drawing of $K_{2,n}$ can achieve constant aspect ratio. Fig. 4.2 shows a straight-line drawing of $K_{2,n}$ with linear area and linear aspect ratio. Notice that a poly-line drawing of the complete bipartite graph $K_{2,n}$ can be thought as a drawing of n paths that start and end at the same two vertices, in the following denoted by a and b, and that do not share any other vertex. In the following we will refer to such paths as to the *paths of* $K_{2,n}$. Precisely, Biedl, Chan, and López-Ortiz proved the following:

Theorem 4.3 *(Biedl* et al. *[BCLO03]) Every planar straight-line drawing of $K_{2,n}$ in a $W \times H$ grid with $W \geq H$ satisfies $W \log H \in \Omega(n)$.*

Corollary 4.1 *(Biedl* et al. *[BCLO03]) Every planar straight-line drawing of $K_{2,n}$ in a $W \times H$ grid satisfies* $\max\{W, H\} \in \Omega(n/\log n)$.

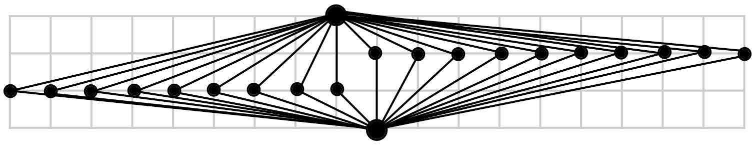


Figure 4.2: A straight-line drawing of $K_{2,n}$ with linear area and linear aspect ratio.

Biedl *et al.* ask whether the $\log H$ factor in Theorem 4.3 can be eliminated and whether the same lower bound holds even in the case of poly-line drawings.

In this chapter we answer both the questions in the affirmative. Namely, we prove the following:

Theorem 4.4 *Every planar straight-line or poly-line drawing of $K_{2,n}$ in a $W \times H$ grid satisfies* $\max\{W, H\} \in \Omega(n)$.

As a main consequence of Theorem 4.4, we obtain a $\Omega(n \log n)$ lower bound on the area requirements of poly-line and straight-line drawings of series-parallel graphs. We remark that no super-linear area lower bound was previously known for poly-line drawings of series-parallel graphs and that $\Omega(\frac{n \log n}{\log \log n})$ was the best known area lower bound for straight-line drawings of series-parallel graphs [Bie05].

Theorem 4.5 *There exist series-parallel graphs requiring $\Omega(n \log n)$ area in any straight-line or poly-line drawing.*

Proof: Consider any series-parallel graph S containing $K_{2,(n/2-2)}$ and a $(n/2)$-node complete ternary tree as subgraphs. A complete ternary tree that has height $h+1$ cannot be drawn on h parallel grid lines [FLW03, Sud04]. Since an n-node complete ternary tree has height $\log_3(2n+1)$, it follows that both sides of the drawing of a $(n/2)$-node complete ternary tree have length $\Omega(\log n)$. Hence, in any straight-line or poly-line drawing of S both sides have length $\Omega(\log n)$ and, by Theorem 4.4, one side has length $\Omega(n)$. The theorem follows. □

Table 4.1 summarizes the area requirements for straight-line and poly-line drawings of series-parallel graphs.

The rest of the chapter is organized as follows. In Sect. 4.2, we study geometric properties of the drawings of $K_{2,n}$; in Sect. 4.3, we prove Theorem 4.4; in Sect. 4.4, we conclude and present some open problems.

	Straight-line				Poly-line			
	UB	ref.	LB	ref	UB	ref.	LB	ref.
Series-Parallel	$O(n^2)$	[dPP90]	$\Omega(n \log n)$	Th. 4.4	$O(n^{1.5})$	[Bie05]	$\Omega(n \log n)$	Th. 4.4

Table 4.1: A table summarizing the area requirements for straight-line and poly-line drawings of series-parallel graphs.

4.2 Lemmata on the Geometry of $K_{2,n}$

In this section we show some lemmata that will be used to prove Theorem 4.4.

Lemma 4.1 *Consider any poly-line drawing of $K_{2,n}$, any path π of $K_{2,n}$, and any vector $\vec{v} = (v_1, v_2)$. There exists a grid point $p \in \pi$ such that $\vec{v} \cdot p \geq \vec{v} \cdot p'$, for any point $p' \in \pi$.*

Proof: If $\vec{v} \cdot a \geq \vec{v} \cdot p'$ or $\vec{v} \cdot b \geq \vec{v} \cdot p'$, for every point $p' \in \pi$, the lemma follows. Otherwise, consider the part π' of π starting at a and ending at the first point p in which $\vec{v} \cdot p \geq \vec{v} \cdot p'$, for every point $p' \in \pi$ (see Fig. 4.3 (a)). Since each point p' of π' has $\vec{v} \cdot p' < \vec{v} \cdot p$, then there exists a small disk D centered at p such that the part of π' enclosed in D is increasing in the direction determined by $\vec{v}$, when π' is oriented from a to p. On the other hand π, when oriented from a to b, cannot be increasing immediately after p in the direction determined by $\vec{v}$, otherwise there would exist a point p'' such that $\vec{v} \cdot p'' > \vec{v} \cdot p$. It follows that π changes slope at p and, by definition of poly-line grid drawing, p is a grid point. □

Lemma 4.2 *Consider any drawing of $K_{2,n}$. Let l be any line that does not intersect or contain the open segment (a, b). There exist no three paths π_1, π_2, and π_3 of $K_{2,n}$ such that: (i) π_1, π_2, and π_3 do not intersect each other; (ii) π_1, π_2, and π_3 are entirely contained in the closed half-plane delimited by l and containing a and b; (iii) each of π_1, π_2, and π_3 touches l at least once.*

Proof: Suppose, for a contradiction, that three paths π_1, π_2, and π_3 of $K_{2,n}$ with the above properties exist. Paths π_1 and π_2 form a cycle $\mathcal{C}$. Line l is external to $\mathcal{C}$ and separates a from b in the exterior of $\mathcal{C}$ (see Fig. 4.3 (b)). Consider any path π_3 between a and b. If π_3 is internal to $\mathcal{C}$, then it can not touch l unless it intersects $\mathcal{C}$. If π_3 is external to $\mathcal{C}$, then it intersects l. If π_3 is part internal and part external to $\mathcal{C}$, then it intersects $\mathcal{C}$. In any case we have a contradiction. □

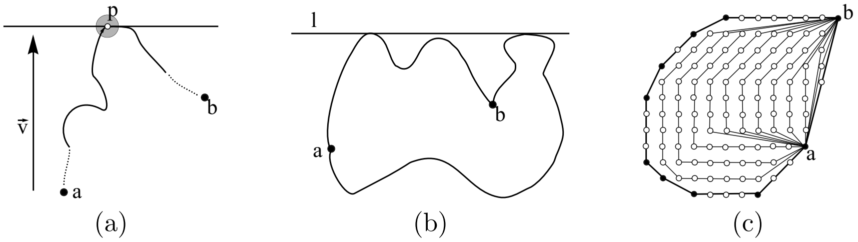


Figure 4.3: (a) Illustration for the proof of Lemma 4.1. Disk D is the small shaded region. (b) Illustration for the proof of Lemma 4.2. (c) Drawing the maximum number of paths in a convex polygon with vertices (drawn as black circles) having integer coordinates.

Let P be any convex polygon in the plane with vertices having integer coordinates. Let G be the set of grid points in the interior or on the border of P. Let a and b be two distinct vertices of P. Let π_1^* and π_2^* be the two paths that connect a and b and that compose P. At least one out of π_1^* and π_2^*, say π_1^*, is different from segment $\overline{ab}$. Let M be the maximum number of paths connecting a and b that can be drawn as non-crossing polygonal paths inside or on the border of P.

Lemma 4.3 *There exists a drawing of M non-crossing polygonal paths connecting a and b such that each path is completely contained inside or on the border of P and one of such paths is drawn as π_1^*.*

Proof: Consider any drawing Γ of M non-crossing polygonal paths connecting a and b and contained inside or on the border of P. If a path of Γ is drawn as π_1^*, there is nothing to prove. Otherwise, observe that no two distinct paths π_i and π_j can pass through points of π_1^*, otherwise π_i and π_j would cross. Hence, Γ has at most one path π passing through points of π_1^*. Remove π from Γ, if π exists, and draw a path in Γ as π_1^*. Since no path different from π passes through a point of π_1^*, the resulting drawing is planar, hence proving the lemma. □

Lemma 4.4 *There exists a drawing of M non-crossing polygonal paths connecting a and b such that each path is completely contained inside or on the border of P and such that one of the paths is represented by segment $\overline{ab}$.*

Proof: We prove the claim by induction on M. If $M = 1$, then drawing a path as segment $\overline{ab}$ proves the claim. Suppose $M \geq 2$. By Lemma 4.3, there exists a drawing Γ of M non-crossing polygonal paths connecting a and b such that each path is inside or on the border of P and one of such paths, say π, is drawn as π_1^*. Remove π from Γ and all the grid points π passes through, except for a and b, from G. Consider the convex closed polygon P' that is the convex hull of the resulting grid point-set G'. The vertices of P' have integer coordinates. Further, P' is such that $M - 1$ non-crossing polygonal paths connecting a and b can be drawn with each path inside or on the border of P'. In fact Γ is a drawing having such a property. Hence, the inductive hypothesis applies and $M - 1$ paths can be drawn so that each path is inside or on the border of P' and so that one of the paths is represented by segment $\overline{ab}$. Considering the drawing of such $M - 1$ paths together with the drawing of π as π_1^* proves the lemma. □

Now assume that a and b are consecutive vertices of P (see Fig. 4.3 (c)). Let G be the set of grid points in the interior or on the border of P. As before, let π_1^* and π_2^* be the two paths that connect a and b and that compose P, where π_1^* is different from segment $\overline{ab}$. Let also M be the maximum number of paths connecting a and b that can be drawn as non-crossing polygonal paths completely contained inside or on the border of P. We iteratively draw paths $\pi_1, \pi_2, \cdots, \pi_N$ connecting a and b inside or on the border of P as follows. Path π_i is drawn when the current convex grid polygon is P_i containing in its interior or on its border a set G_i of grid points. At the first step $P_1 = P$ and $G_1 = G$. If P_i does not coincide with segment $\overline{ab}$, draw path π_i as the polygonal line that connects a and b, that lies on P_i, and that is different from segment $\overline{ab}$. Remove the grid points that lie on P_i, except for a and b, from G_i, obtaining a new set of grid points G_{i+1}. Then, P_{i+1} is the convex hull of G_{i+1}. If P_i coincides with segment $\overline{ab}$, draw the path π_i as segment $\overline{ab}$. We observe the following:

Lemma 4.5 *Paths $\pi_1, \pi_2, \cdots, \pi_N$ are non-crossing polygonal lines that connect a and b and that completely lie inside or on the border of P. Further, $N = M$.*

Proof: The first part of the statement is trivial. We prove that $N = M$ by induction on M. If $M = 2$, then the claim trivially holds, since π_1 is drawn as π_1^* and π_2 as $\overline{ab}$. Suppose $M \geq 3$. By Lemma 4.3, there exists a drawing Γ of M non-crossing polygonal paths connecting a and b such that each path is completely contained inside or on the border of P and one of such paths, say π_1, is drawn as π_1^*. Remove π_1 from Γ and all the grid points π_1 passes through from G. Consider the convex closed polygon P' that is the convex hull of the resulting grid point-set G'. Clearly, the vertices of P' have integer coordinates. Further, P' is such that $M-1$ non-crossing polygonal paths connecting a and b can be drawn such that each path is completely contained inside or on the border of P'. In fact Γ is a drawing having such a property. Hence, the inductive hypothesis applies and the drawing algorithm described before the statement of the lemma draws $M-1$ non-crossing polygonal paths inside or on the border of P'. Considering such paths together with the drawing of π_1 as π_1^* proves the lemma. □

4.3 A Lower Bound on the Area Requirements of $K_{2,n}$

In this section we prove Theorem 4.4. By definition, a straight-line drawing is also a poly-line drawing. Hence, it suffices to prove Theorem 4.4 for poly-line drawings. Consider any poly-line drawing of $K_{2,n}$. Let R be the minimum closed axis-parallel rectangle enclosing a and b (see Fig. 4.4). Let $l_{a,b}$ be the line through a and b. Suppose, w.l.o.g., that $y(a) \leq y(b)$. Suppose also that the slope of $l_{a,b}$ is greater or equal than 0, the case in which the slope of $l_{a,b}$ is less than 0 being analogous. Let c and d be the upper left corner and the lower right corner of R, respectively. Let h_a and v_a (h_b and v_b) be the horizontal and vertical lines through a (resp. through b), respectively. For any line l, denote by $H^+(l)$ (resp. by $H^-(l)$) the closed half-plane delimited by l and containing the normal vector of l increasing in the y-direction (resp. decreasing in the y-direction). If l is a vertical line, then $H^+(l)$ (resp. $H^-(l)$) denotes the closed half-plane delimited by l and containing the normal vector of l increasing in the x-direction (resp. decreasing in the x-direction). Let d_1 and d_2 be the horizontal and vertical distance between a and b, respectively. The width W and the height H of the drawing are such that $W \geq d_1$ and $H \geq d_2$.

Consider the half-plane $H^+(h_b)$. By Lemma 4.1 with $\vec{v} = (0,1)$, for each path π intersecting $H^+(h_b)$, there exists a grid point $p \in \pi$ whose y-coordinate is maximum among the points of π. Clearly, p belongs to $H^+(h_b)$. Hence, p belongs to an horizontal grid line l that does not intersect or contain the open segment (a,b). By Lemma 4.2, at most two paths of $K_{2,n}$ have their points with greatest y-coordinate belonging to l. It follows that, if a linear number of paths of $K_{2,n}$ intersects $H^+(h_b)$, then their points with greatest y-coordinate belong to a linear number of distinct horizontal grid lines and hence $H \in \Omega(n)$.

Similar arguments show that, if a linear number of edges intersect $H^-(h_a)$, $H^+(v_b)$, or $H^-(v_a)$, then $H \in \Omega(n)$, $W \in \Omega(n)$, or $W \in \Omega(n)$, respectively. If

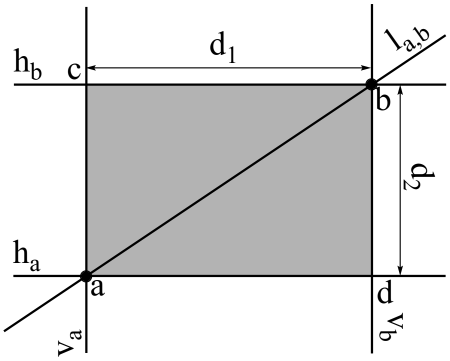


Figure 4.4: (a) Illustration of the notation for the proof of Theorem 4.4.

there exists no linear number of edges intersecting $H^+(h_b)$, $H^-(h_a)$, $H^+(v_b)$, or $H^-(v_a)$, then a linear number of edges is completely inside R. We show that this implies that $\max\{d_1, d_2\} \in \Omega(n)$, and hence that $\max\{W, H\} \in \Omega(n)$.

Let M be the maximum number of paths of $K_{2,n}$ that can be drawn inside R. By Lemma 4.4, there exists a drawing of M paths connecting a and b, and completely lying inside R, such that one of the paths is drawn as segment $\overline{ab}$. Since $M \in \Omega(n)$, then either a linear number of paths of $K_{2,n}$ is contained in the triangle T_1 having a, b, and c as vertices, or a linear number of paths of $K_{2,n}$ is contained in the triangle T_2 having a, b, and d as vertices. Suppose that a linear number of paths is contained into T_1, the other case being symmetric.

Let $M_1 \in \Omega(n)$ be the maximum number of paths of $K_{2,n}$ that can be drawn inside T_1, and let G_1 be the set of grid points inside or on the border of T_1. By Lemma 4.5, a sequence of M_1 non-crossing paths $\Pi = (\pi_1, \pi_2, \cdots, \pi_{M_1})$ connecting a and b and completely inside or on the border of T_1 can be drawn by repeating the following two operations, for $1 \leq i < M_1$: (1) consider the current convex grid polygon P_i (when $i = 1$ then $P_1 = T_1$); let G_i be the set of grid points inside or on the border of P_i; draw path π_i as the part of P_i that connects a and b, and that is different from segment $\overline{ab}$; (2) delete from G_i the grid points π_i passes through, obtaining a set of grid points G_{i+1}. Closed convex polygon P_{i+1} is the convex hull of G_{i+1}. Path π_{M_1} is drawn as segment $\overline{ab}$. See Fig. 4.5.

In order to prove that $M_1 \in \Omega(n)$ implies $\max\{d_1, d_2\} \in \Omega(n)$, we study paths $\pi_1, \pi_2, \cdots, \pi_{M_1}$. Such a study reveals interesting properties of the grid

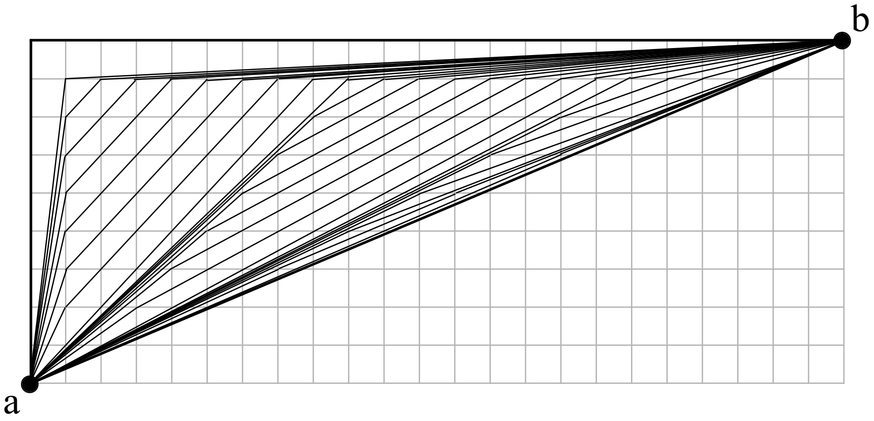


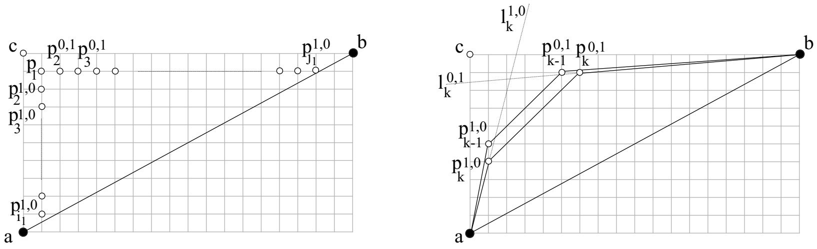
Figure 4.5: Paths $\pi_1, \pi_2, \cdots, \pi_{M_1}$ in Π.

that we skecth here and detail in the following. First, we observe that each path in Π is composed by two or three segments, i.e., each path has one or two bends. A sequence of paths that are consecutive in Π and that are each composed by three segments is such that all the "second segments" of the paths have the same slope. We show that, in a sequence of paths such that the second segments of the paths have the same slope, all the bends lie on two lines, having slopes one greater and one less than the slope of segment $\overline{ab}$. The more sequences of three-segments-paths that are consecutive in Π are considered, the more the slope of the first, of the second, and of the third segment of the paths approaches to the slope of segment $\overline{ab}$. Consider a sequence of paths such that the second segments of the paths have the same slope $\frac{s_1}{s_2}$. Then, the bends of such paths lie on two lines with slopes, say, $\frac{s_3}{s_4}$ and $\frac{s_5}{s_6}$, such that $s_3 + s_5 = s_1$ and $s_4 + s_6 = s_2$. Further, the next sequence of paths whose second segments have the same slope is such that the bends of such paths lie on two lines with slopes $\frac{s_1}{s_2}$ and $\frac{s_3}{s_4}$ (or $\frac{s_1}{s_2}$ and $\frac{s_5}{s_6}$), and the second segments of such paths have slope $\frac{s_1+s_3}{s_2+s_4}$ (resp. $\frac{s_1+s_5}{s_2+s_6}$). We subdivide Π into disjoint subsequences $\Pi_1, \Pi_2, \cdots, \Pi_f$ and we argue that Π_1 has at most $\max\{d_1, d_2\}$ paths and that Π_i has at most $\max\{d_1, d_2\}/2^{i-2}$ paths, for $2 \leq i \leq f$; such bounds lead to conclude that, as long as $M_1 \in \Omega(n)$, $\max\{d_1, d_2\} \in \Omega(n)$.

Path π_1 is composed of segments $\overline{ac}$ and $\overline{cb}$. Let p_1 be the point one vertical unit below and one horizontal unit to the right of c. Consider the following two sequences of grid points (see Fig. 4.6 (a)). Sequence $S_{1,0}$ is composed of points:

$$\begin{aligned}
p_1^{1,0} &= p_1,\\
p_2^{1,0} &= (x(p_1), y(p_1) - 1),\\
p_3^{1,0} &= (x(p_1), y(p_1) - 2),\\
&\cdots,\\
p_{i_1}^{1,0} &= (x(p_1), y(p_1) - (i_1 - 1)),
\end{aligned}$$

where i_1 is the maximum such that point $(x(p_1), y(p_1) - (i_1 - 1))$ is contained inside T_1. Sequence $S_{0,1}$ is composed of points:



(a) (b)

Figure 4.6: (a) Sequences $S_{1,0}$ and $S_{0,1}$. (b) Path π_{k+1} is a polygonal line composed of segments $\overline{ap_k^{1,0}}$, $\overline{p_k^{1,0}p_k^{0,1}}$, $\overline{p_k^{0,1}b}$, for $k = 1, 2, \cdots, \min\{i_1, j_1\}$.

$$\begin{aligned}
p_1^{0,1} &= p_1,\\
p_2^{0,1} &= (x(p_1) + 1, y(p_1)),\\
p_3^{0,1} &= (x(p_1) + 2, y(p_1)),\\
&\cdots,\\
p_{j_1}^{0,1} &= (x(p_1) + (j_1 - 1), y(p_1)),
\end{aligned}$$

where j_1 is the maximum such that point $(x(p_1) + (j_1 - 1), y(p_1))$ is contained inside T_1. Notice that the points of $S_{1,0}$ lie on a line with slope $\frac{1}{0} = \infty$ and the points of $S_{0,1}$ lie on a line with slope $\frac{0}{1} = 0$. In the following, we show that a subsequence Π_1 of Π, starting at π_2 and composed of paths consecutive in Π, "consumes" the points in $S_{1,0}$ and in $S_{0,1}$, i.e., each point in $S_{1,0}$ and each point in $S_{0,1}$ is traversed by a path in Π_1; further, each path in Π_1 passes through a distinct point of the one of $S_{1,0}$ and $S_{0,1}$ that has the greater number of points.

We claim that path π_{k+1} is a polygonal line composed of segments $\overline{ap_k^{1,0}}$, $\overline{p_k^{1,0}p_k^{0,1}}$, $\overline{p_k^{0,1}b}$, for $k = 1, 2, \cdots, \min\{i_1, j_1\}$ (notice that $p_1^{1,0} = p_1^{0,1} = p_1$, hence π_2 is composed by only two segments). The claim directly implies that the second segment of path π_{k+1}, for $k = 2, 3, \cdots, \min\{i_1, j_1\}$, has slope $\frac{1}{1} = 1$. We prove the claim by induction on k. Let $l_k^{1,0}$, $l_k^{1,1}$, and $l_k^{0,1}$, be the lines through a and $p_k^{1,0}$, through $p_k^{1,0}$ and $p_k^{0,1}$, and through $p_k^{0,1}$ and b, respectively.

In the base case $k = 1$. Observe that $H^+(l_1^{1,0})$ and $H^+(l_1^{0,1})$ do not contain grid points that are inside or on the border of T_1, and that do not belong to π_1, except for p_1. Hence, π_2 is composed by segments $\overline{ap_1}$ and $\overline{p_1b}$, proving the claim in the base case. Suppose that the claim holds for paths $\pi_2, \pi_3, \cdots, \pi_k$. Then, π_k is a polygonal line composed of segments $\overline{ap_{k-1}^{1,0}}$, $\overline{p_{k-1}^{1,0}p_{k-1}^{0,1}}$, $\overline{p_{k-1}^{0,1}b}$. We prove that the claim holds for path π_{k+1} (see Fig. 4.6 (b)). It is easy to see that $H^+(l_k^{1,0})$ and $H^+(l_k^{0,1})$ do not contain grid points internal to polygon $\pi_k \cup \overline{ab}$, except for $p_k^{1,0}$ and for $p_k^{0,1}$, respectively. Further, no grid point is contained inside quadrilateral $(p_{k-1}^{0,1}, p_{k-1}^{1,0}, p_k^{1,0}, p_k^{0,1})$. Namely, any grid point of the plane lies on a line with slope $\frac{1}{1}$, and the line that has slope $\frac{1}{1}$, that contains grid points internal to $\pi_k \cup \overline{ab}$, and that is closer to $l_{k-1}^{1,1}$ is line $l_k^{1,1}$ through $p_k^{1,0}$ and $p_k^{0,1}$. Hence, path π_{k+1} is composed by segments $\overline{ap_k^{1,0}}$, $\overline{p_k^{1,0}p_k^{0,1}}$, $\overline{p_k^{0,1}b}$, proving the claim.

Three cases have to be considered, namely $i_1 = j_1$, $i_1 < j_1$, and $i_1 > j_1$. If $i_1 = j_1$, we claim that there is no grid point internal to polygon $\pi_{i_1+1} \cup \overline{ab}$. Observe that, since a is one unit to the left of the vertical line on which the points of $S^{1,0}$ lie, and since b is one unit above the horizontal line on which the points of $S^{0,1}$ lie, then, if there is any grid point internal to polygon $\pi_{i_1+1} \cup \overline{ab}$, either point $p_{i_1+1}^{1,0} = (x(p_1), y(p_1) - i_1)$ or $p_{j_1+1}^{0,1} = (x(p_1) + j_1, y(p_1))$ is internal to $\pi_{i_1+1} \cup \overline{ab}$ (see Fig. 4.7 (a)). However, by the maximality of i_1 and j_1, both $p_{i_1+1}^{1,0}$ and $p_{j_1+1}^{0,1}$ are outside or on the border of T_1, and hence they are not internal to $\pi_{i_1+1} \cup \overline{ab}$. Since there is no grid point internal to polygon $\pi_{i_1+1} \cup \overline{ab}$, then the only path of $K_{2,n}$ after $\pi_{i_1+1} = \pi_{j_1+1}$ in Π is segment $\overline{ab}$.

Now consider the case in which $i_1 < j_1$ (the case in which $i_1 > j_1$ is analogous). Sequence $S_{1,0}$ is "over", i.e., there is a path π_i passing through each point of $S_{1,0}$. Let $S_{1,1}$ be the sequence defined as follows (see Fig. 4.7 (b)):

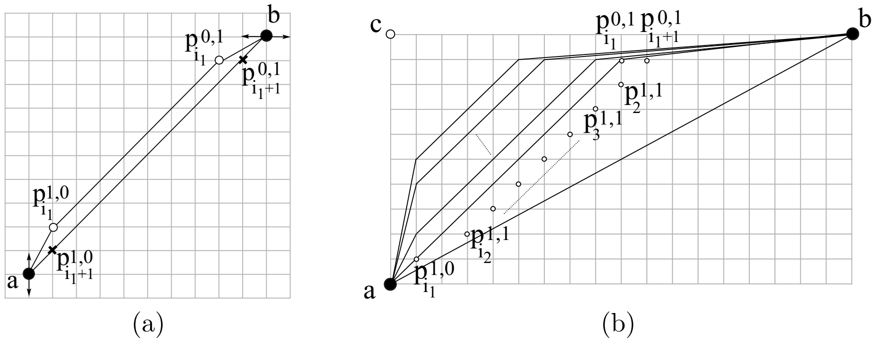


Figure 4.7: (a) When $i_1 = j_1$, no grid point is internal to $\pi_{i_1+1} \cup \overline{ab}$. (b) Sequence $S_{1,1}$.

$$
\begin{aligned}
&p_1^{1,1} = p_{i_1+1}^{0,1},\\
&p_2^{1,1} = (x(p_{i_1+1}^{0,1}) - 1, y(p_{i_1+1}^{0,1}) - 1),\\
&p_3^{1,1} = (x(p_{i_1+1}^{0,1}) - 2, y(p_{i_1+1}^{0,1}) - 2),\\
&\cdots,\\
&p_{i_2}^{1,1} = (x(p_{i_1+1}^{0,1}) - (i_2 - 1), y(p_{i_1+1}^{0,1}) - (i_2 - 1)),
\end{aligned}
$$

where i_2 is the maximum such that point $((x(p_{i_1+1}^{0,1}) - (i_2-1), y(p_{i_1+1}^{0,1}) - (i_2 - 1))$ is contained inside T_1. Sequence $S_{1,1}$ “replaces” sequence $S_{1,0}$, namely path π_{i_1+k+1}, with $1 \leq k \leq \min\{i_2, j_1 - i_1\}$, is a polygonal line composed of segments $\overline{ap_k^{1,1}}$, $\overline{p_k^{1,1}p_{i_1+k}^{0,1}}$, $\overline{p_{i_1+k}^{0,1}b}$ (the proof of such a claim is analogous to the proof that π_{k+1} is composed by segments $\overline{ap_k^{1,0}}$, $\overline{p_k^{1,0}p_k^{0,1}}$, $\overline{p_k^{0,1}b}$, for $k = 1, 2, \cdots, \min\{i_1, j_1\}$). Notice that the bends of such paths lie on two lines with slope $\frac{1}{1} = 1$ and $\frac{0}{1} = 0$, while the second segments of such paths lie on lines with slope $\frac{1+0}{1+1} = \frac{1}{2}$.

Again, three cases have to be considered. In the first case, we have $i_2 = j_1 - i_1$. Hence, path π_{j_1+1} passes through the last point of $S_{1,1}$ and through the last point of $S_{0,1}$. Then no grid point lies inside polygon $\pi_{j_1+1} \cup \overline{ab}$ (the proof of such a claim is analogous to the one that there is no grid point internal to polygon $\pi_{i_1+1} \cup \overline{ab}$ when $i_1 = j_1$), and hence the only path of $K_{2,n}$ after $\pi_{i_1+i_2+1} = \pi_{j_1+1}$ in Π is segment $\overline{ab}$. Otherwise, one of the two sequences $S_{1,1}$

and $S_{1,0}$ ends before the other. Suppose that sequence $S_{1,1}$ ends before $S_{0,1}$. Then $S_{1,1}$ is replaced by a sequence $S_{1,2}$ of points lying on a line with slope $\frac{1}{2}$. Namely, such points have coordinates:

$$p_k^{1,2} = \Big(x(p^{0,1}_{i_1+i_2+1}) - 2(k-1), x(p^{0,1}_{i_1+i_2+1}) - (k-1)\Big),$$

for $1 \leq k \leq i_3$, where i_3 is the maximum index such that point $(x(p^{0,1}_{i_1+i_2+1}) - 2(i_3-1), x(p^{0,1}_{i_1+i_2+1}) - (i_3-1))$ is internal to T_1. Each path $\pi_{i_1+i_2+k+1}$, with $1 \leq k \leq \min\{i_3, j_1 - i_1 - i_2\}$, passes through point $p_k^{1,2}$ and through point $p^{0,1}_{i_1+i_2+k}$, and the second segment of each of such paths has slope $\frac{1+0}{2+1} = \frac{1}{3}$.

The above argument iterates while sequence $S_{0,1}$ is not over, i.e., while there are points of $S_{0,1}$ that are not traversed by paths in Π. From the above discussion, all paths that come after π_1 in Π pass through distinct points of $S_{0,1}$, while sequence $S_{0,1}$ is not over. Hence, supposing $i_1 \leq j_1$ (the case in which $j_1 \leq i_1$ being analogous), $\Pi_1 = (\pi_2, \pi_3, \cdots, \pi_{j_1+1})$ is the desired subsequence of Π passing through all points of $S_{1,0}$ and through all points of $S_{0,1}$. Further, there exists an index $l \geq 1$ such that: (1) every point in $S_{1,i}$ is traversed by a path in Π_1, for $0 \leq i < l$, and (2) some points of $S_{1,l}$ are eventually traversed by a path in Π_1.

After drawing path π_{j_1+1} (that passes through the last point of $S^{0,1}$), either sequence $S_{1,l}$ is simultaneously over, i.e., $j_1 = i_1 + i_2 + \cdots + i_l$, or $S_{1,l}$ still contains points internal to T_1 and not traversed by any path in Π_1. In the former case we have that no grid point is internal to polygon $\pi_{j_1+1} \cup \{ab\}$ and hence the only path of $K_{2,n}$ after π_{j_1+1} in Π is segment $\overline{ab}$. In the latter case, more than one path could exist in Π after π_{j_1+1}. Namely, sequence $S_{0,1}$ is now replaced by a sequence $S_{1,l+1}$, whose points lie on a line with slope $\frac{0+1}{1+l} = \frac{1}{l+1}$ passing through the first point of $S_{1,l}$ that is not traversed by a path in Π_1 (see Fig. 4.8).

The whole argument is now repeated again. Namely, we search for a subsequence Π_2 of Π such that Π_2 "consumes" the points in $S_{1,l}$ and the points in $S_{1,l+1}$, i.e., such that each point in $S_{1,l}$ that is not traversed by a path in Π_1 is traversed by a path in Π_2 and such that each point in $S_{1,l+1}$ is traversed by a path in Π_2; further, each path in Π_2 passes through a point of the one out of $S_{1,l}$ and $S_{1,l+1}$ that has the greater number of points. Again, Π_2 is generally found in several steps, where at each step two sequences S_{y_1,x_1} and S_{y_2,x_2} of grid points are considered (at the first step such sequences are $S_{1,l}$ and $S_{1,l+1}$, where the points of $S_{1,l}$ that are traversed by paths in Π_1 are not considered). At each step, the smallest between S_{y_1,x_1} and S_{y_2,x_2} is consumed by the paths

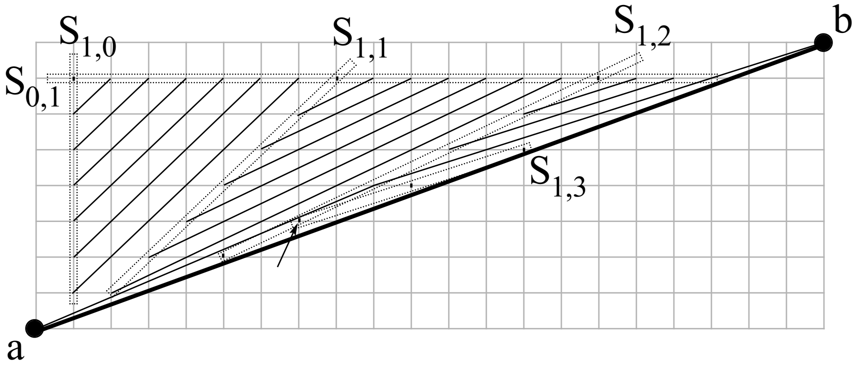


Figure 4.8: The case in which sequence $S_{1,l}$ still contains at least one point after drawing path π_{j_1+1}. In this example $l = 2$. To improve the readability of the drawing, only the second segments of the paths in Π_1 are drawn, but for path π_{j_1+1}, which is entirely drawn. The arrow shows the first point of $S_{1,2}$ that is not traversed by a path in Π_1.

in a subsequence of Π_2 and is replaced by a sequence of points lying on a line with slope $\frac{y_1+y_2}{x_1+x_2}$, hence starting a new step. After a certain number of steps, all points in $S_{1,l}$ and in $S_{1,l+1}$ are traversed by a path in Π_2. When the last path of Π_2 is drawn, either the currently considered sequences $S_{y_1^*,x_1^*}$ and $S_{y_2^*,x_2^*}$ are simultaneously over, or there are still points, not traversed by any path in Π_2, in the one out of $S_{y_1^*,x_1^*}$ and $S_{y_2^*,x_2^*}$ that is different from both $S_{1,l}$ and in $S_{1,l+1}$. In the former case, no grid point is inside the polygon composed of the last drawn path and of $\overline{ab}$, and hence the only path of $K_{2,n}$ after the last path of Π_2 in Π is segment $\overline{ab}$. In the latter case, the one between $S_{y_1^*,x_1^*}$ and $S_{y_2^*,x_2^*}$ that is over, say $S_{y_2^*,x_2^*}$, is replaced by a sequence $S_{y_1^*+y_2^*,x_1^*+x_2^*}$, whose grid points lie on a line with slope $\frac{y_1^*+y_2^*}{x_1^*+x_2^*}$, and the whole argument is repeated again, searching for a subsequence Π_3 of Π such that Π_3 consumes the points in $S_{y_1^*,x_1^*}$ and the points in $S_{y_1^*+y_2^*,x_1^*+x_2^*}$. Clearly, there exists an index f such that $\Pi = \{\pi_1\} \cup \Pi_1 \cup \Pi_2 \cup \cdots \Pi_f \cup \{\overline{ab}\}$.

We now compute how many paths exist in Π, as a function of d_1 and d_2. Denote by $S_{y_1^i,x_1^i}$ and by $S_{y_2^i,x_2^i}$ the sequences of grid points that are consumed by Π_i, where the grid points in $S_{y_1^i,x_1^i}$ lie on a line with slope y_1^i/x_1^i and the grid points in $S_{y_2^i,x_2^i}$ lie on a line with slope y_2^i/x_2^i. Notice that, with the above notation, $S_{y_1^1,x_1^1} = S_{1,0}$ and $S_{y_2^1,x_2^1} = S_{0,1}$, and, if $i_1 \leq j_1$, $S_{y_1^2,x_1^2} = S_{1,l}$, and

$S_{y_2^2,x_2^2} = S_{1,l+1}$. It is easy to prove that $x_1^i, y_1^i, x_2^i, y_2^i \geq 2^{i-2}$, for $i \geq 2$. Notice that we already observed that such a claim holds when $i = 2$. From the above discussion, we have that y_1^i is obtained as the sum of the numerators y_a^{i-1} and y_b^{i-1} of the slopes of two lines containing grid points traversed by paths in Π_{i-1}. Inductively, $y_a^{i-1}+y_b^{i-1} \geq y_1^{i-1}+y_2^{i-1} \geq 2^{i-3}+2^{i-3} \geq 2^{i-2}$. Analogously $y_2^i, x_1^i, x_2^i \geq 2^{i-2}$.

The number of paths in Π_i is the number of grid points in the one out of $S_{y_1^i,x_1^i}$ and $S_{y_2^i,x_2^i}$ with the greater number of points. When $i = 1$, each of $S_{1,0}$ and $S_{0,1}$ has at most $\max\{d_1, d_2\}$ grid points. Further, for $i \geq 2$, $S_{y_1^i,x_1^i}$ and $S_{y_2^i,x_2^i}$ lie on lines with slopes whose numerators and denominators are greater or equal than 2^{i-2}. Hence, each of such sequences has at most $\frac{\max\{d_1,d_2\}}{2^{i-2}} + 1$ grid points. Hence, the total number of paths in Π is at most

$$\underbrace{1}_{\pi_1} + \underbrace{\max\{d_1, d_2\}}_{\text{paths in } \Pi_1} + \underbrace{\sum_{i=2}^{f} \left(\frac{\max\{d_1, d_2\}}{2^{i-2}} + 1 \right)}_{\text{paths in } \Pi_i, \text{ for } 2 \leq i \leq f} + \underbrace{1}_{\overline{ab}} \leq$$

$$\max\{d_1, d_2\} + 2\max\{d_1, d_2\} + \log_2(\max\{d_1, d_2\}) + 2 < 4\max\{d_1, d_2\} + 2.$$

Since the number of paths in Π is $\Omega(n)$, then $\max\{d_1, d_2\} \in \Omega(n)$ and hence $\max\{W, H\} \in \Omega(n)$. Theorem 4.4 follows.

4.4 Conclusions and Open Problems

In this chapter we have shown that there exist series-parallel graphs requiring $\Omega(n \log n)$ area in any straight-line or poly-line grid drawing. As far as we know the best upper bound for the area requirements of poly-line drawings of series-parallel graphs is $O(n^{3/2})$ [Bie05], while no sub-quadratic area upper bound is known in the case of straight-line drawings. Hence, in both cases, the gap between the upper and the lower bound is large, and the following problems are quite natural:

Open Problem 4.1 *Which are the asymptotic bounds for the area requirements of straight-line planar drawings of series-parallel graphs?*

Open Problem 4.2 *Which are the asymptotic bounds for the area requirements of poly-line planar drawings of series-parallel graphs?*

Chapter 5

Straight-line Drawings of Outerplanar Graphs

In this chapter[1] we consider straight-line and poly-line drawings of outerplanar graphs in small area. We show four linear-time algorithms for constructing planar straight-line grid drawings of outerplanar graphs. The first and the second algorithm are for balanced outerplanar graphs. Both require linear area. The drawings produced by the first algorithm are not outerplanar while those produced by the second algorithm are. On the other hand, the first algorithm constructs drawings with better angular resolution. The third and the fourth algorithm construct outerplanar drawings of general outerplanar graphs with $O(n^{1.48})$ area and with $O(dn \log n)$ area, where d is the degree of the graph. Further, we study the interplay between the area requirements of the drawings of an outerplanar graph and the area requirements of a special class of drawings of its dual tree. Finally, we settle in the negative a conjecture [Bie02] on the area requirements of outerplanar graphs by showing that snowflake graphs admit linear-area drawings.

5.1 Introduction

Outerplanar graphs are one of the classes of graphs that has attracted more research interest in the field of Graph Drawing. For example, it has been proved by Gritzmann, Mohar, Pach, and Pollack in [GPP91] that the outer-

[1] The contents of this chapter are a joint work with Giuseppe Di Battista, appeared in [DF05, Fra07b] and to appear in [DF09].

planar graphs are exactly the graphs that have a planar straight-line embedding into every point set. Bose [Bos02] showed an $O(n \log^3 n)$ algorithm to compute such an embedding. As an other (more graph-theoretical) example, Gonçalves [Gon05] proved that the edges of any planar graph can be partitioned into two sets, each set inducing an outerplanar graph.

Determining the area requirements of outerplanar graphs has been considered an intriguing challenge for almost thirty years. In 1981, Dolev and Trickey [DT81] showed that every n-vertex outerplanar graph whose degree is bounded by four admits a poly-line drawing in $\Theta(n)$ area. The techniques presented in [DT81] can be modified in order to obtain poly-line drawings of outerplanar graphs with degree d in $O(d^2 n)$ area, as pointed out in [Bie02]. More recently, the problem of obtaining minimum-area drawings of outerplanar graphs has been tackled by Biedl, who first provided a sub-quadratic area upper bound for poly-line drawings of general outerplanar graphs. Namely, she proved that outerplanar graphs admit poly-line drawings in $O(n \log n)$ area. Such a result, which first appeared in [Bie02], can be obtained as a Corollary of the results of the same author on series-parallel graphs, appeared in [Bie05] and sketched in Chapter 4. In fact, outerplanar graphs are series-parallel graphs with fan-out equal to two. Hence, Theorem 4.2 applies to such graphs, providing an $O(n \log n)$ area upper bound. In the same two papers, Biedl [Bie02, Bie05] conjectured that there exists a class of outerplanar graphs called "snowflake graphs" requiring $\Omega(n \log n)$ area in any planar straight-line or poly-line drawing.

Concerning straight-line drawings, it can be noticed as preliminary observations that: (i) any strictly-convex polygon can be used as drawing of the outer face of the outerplanar graph; adding the remaining edges of the graph preserves the planarity of the drawing; however, the size of a grid containing a n-vertex strictly-convex polygon is $\Omega(n^3)$ [And63, BP92, BT04, Rab93], hence no area bound better than $O(n^3)$ can be obtained for straight-line grid drawings of outerplanar graphs by relying on the *a priori* construction of a set of points in convex position; (ii) any set of grid points in general position allows for constructing straight-line embeddings of outerplanar graphs [GPP91, Bos02]; however, the size of a grid containing n points in general position is $\Omega(n^2)$, hence no area bound better than $O(n^2)$ can be obtained for straight-line grid drawings of outerplanar graphs by relying on the *a priori* construction of a set of points in general position.

The first interesting bound for straight-line drawings of outerplanar graphs appeared in [GR03a] (further published in [GR07]). Garg and Rusu showed that every n-vertex outerplanar graph with degree d has a straight-line drawing with $O(dn^{1.48})$ area. Such a result is achieved by means of an algorithm that works by induction on the dual tree T of the outerplanar graph G, namely, it finds a path P in T, it removes from G the subgraph G_P that has P as dual, it inductively draws the outerplanar graphs that are disconnected by such a removal, and it puts all the drawings of such outerplanar graphs together with a drawing of G_P, obtaining a drawing of the whole outerplanar graph.

In this chapter we study straight-line drawings of outerplanar graphs and we show the following results.

- We show a linear-time algorithm for drawing a balanced outerplanar graph in linear area and with angular resolution greater than $\frac{c}{\sqrt{n}}$, where c is a constant. A balanced outerplanar graph is such that its dual tree is a balanced tree. The drawings obtained with such an algorithm are not outerplanar.

- We define a new type of drawings of binary trees, called star-shaped drawings. We study the interplay between the drawings of an outerplanar graph and the star-shaped drawings of its dual tree. Namely, we show that, given a drawing of an outerplanar graph it is possible to find a star-shaped drawing of its dual tree with the same area bound. Conversely, given a star-shaped drawing of a binary tree it is possible to find a drawing of its dual outerplanar graph with the same area bound, but for the placement of two special vertices.

- Based on the above correspondence, we show a linear-time algorithm for drawing a balanced outerplanar graph in linear area. The drawings obtained with such an algorithm are outerplanar, but the angular resolution is worse with respect to those computed by the previous algorithm.

- We show a linear-time algorithm for constructing outerplanar drawings of general outerplanar graphs with $O(n^{1.48})$ area. Such an algorithm is based on the correspondence between the drawings of an outerplanar graph and the star-shaped drawings of its dual tree, on a decomposition technique of binary trees presented in [Cha02], and on a simple geometric construction of star-shaped drawings of binary trees.

- We show a linear-time algorithm for constructing outerplanar drawings of general outerplanar graphs with $O(dn \log n)$ area. Such an algorithm is based on the correspondence between the drawings of an outerplanar graph and the star-shaped drawings of its dual tree, and on a quite involved geometric construction of star-shaped drawings of binary trees.

- We show a linear-time algorithm for constructing straight-line drawings of snowflake graphs in $O(n)$ area, settling in the negative the above cited conjecture appeared in [Bie02].

In all the above algorithmic results, we deal with maximal outerplanar graphs. However, we remark that this is not a limitation since that any outerplanar graph can be always augmented in linear time with a linear number of dummy edges to a maximal outerplanar graph. Hence, an algorithm for drawing an (even disconnected) outerplanar graph G can first augment G to a maximal outerplanar graph G' by adding dummy edges, can then draw G', and can finally remove the dummy edges inserted at the first step, thus obtaining a drawing of G.

Table 5.1 summarizes the area requirements for straight-line and poly-line drawings of outerplanar graphs.

	Straight-line				**P**oly-line			
	UB	**r**ef.	**L**B	**r**ef	**U**B	**r**ef.	**L**B	**r**ef.
Outerpl.	$O(n^{1.48})$	Th. 5.5	$\Omega(n)$	*trivial*	$O(n \log n)$	[Bie05]	$\Omega(n)$	*trivial*
Degree-d Outerpl.	$O(dn \log n)$	Th. 5.6	$\Omega(n)$	*trivial*	$O(dn \log n)$ $O(d^2 n)$	Th. 5.6 [Bie05]	$\Omega(n)$	*trivial*
Balanced Outerpl.	$O(n)$	Th. 5.3	$\Omega(n)$	*trivial*	$O(n)$	Th. 5.3	$\Omega(n)$	*trivial*

Table 5.1: A table summarizing the area requirements for straight-line and poly-line drawings of outerplanar graphs.

The rest of the chapter is organized as follows. In Sect. 5.2, we study non-outerplanar drawings of balanced outerplanar graphs; in Sect. 5.3, we study the relationships between straight-line drawings of outerplanar graphs and star-shaped drawings of binary trees; in Sect. 5.4, we study outerplanar drawings of balanced outerplanar graphs; in Sect. 5.5 and 5.6, we study outerplanar drawings of outerplanar graphs; finally, in Sect. 5.7, we study outerplanar drawings of snowflake graphs, we conclude and present some open problems.

5.2 Non-Outerplanar Drawings of Complete and Balanced Outerplanar Graphs

In this section we present an algorithm for constructing linear-area non-outerplanar drawings of complete and balanced outerplanar graphs. A paper [Bie02] by Biedl reports that Biedl and Demaine have observed that every outerplanar graph whose dual tree has diameter k has a straight-line drawing in $O(kn)$ area. Balanced trees are trees whose diameter is $O(\log n)$, hence their bound here reads as $O(n \log n)$.

We call G_h a complete outerplanar graph with height h and Γ_h a planar straight-line grid drawing of G_h. Let also u_h, v_h and w_h be the left vertex, the right vertex and the central vertex of G_h, respectively (see Sect. 1.4).

We show an inductive algorithm to draw complete outerplanar graphs. It draws Γ_h exploiting Γ_{h-1}. The algorithm is as follows.

- ***Base case:*** if $h = 1$, then place u_1 in $(0, 0)$, v_1 in $(1, 1)$ and w_1 in $(1, 0)$.

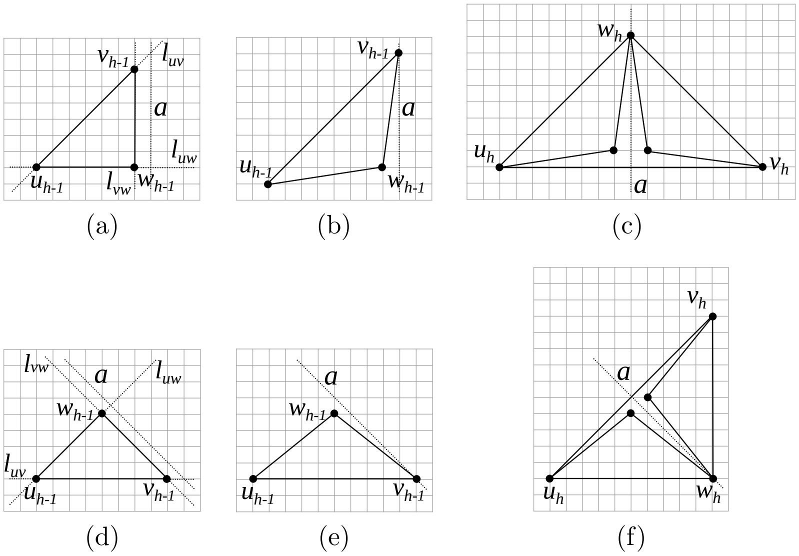


Figure 5.1: Inductive case of the algorithm. (a)–(c) h even: (a) Γ_{h-1}; (b) Γ_{h-1} after the shifts of u_{h-1} and v_{h-1}; (c) Γ_h. (d)–(f) h odd: (d) Γ_{h-1}; (e) Γ_{h-1} after the shifts of u_{h-1} and v_{h-1}; (f) Γ_h.

- ***Inductive case:*** if $h > 1$, inductively suppose to have a drawing Γ_{h-1} of G_{h-1} such that: (i) If $h - 1$ is odd, then the convex hull of Γ_{h-1} is an isosceles rectangular triangle whose vertices are u_{h-1}, v_{h-1}, and w_{h-1}; the catheti (u_{h-1}, w_{h-1}) and (v_{h-1}, w_{h-1}) lie on lines parallel to the axes, so that u_{h-1} and w_{h-1} lie on the same horizontal line l_{uw}, with u_{h-1} to the left of w_{h-1}, and so that v_{h-1} and w_{h-1} lie on the same vertical line

l_{vw}, with v_{h-1} above w_{h-1}; the hypothenuse lies on a line l_{uv} with slope $\pi/4$. (ii) If $h-1$ is even, then the convex hull of Γ_{h-1} is an isosceles rectangular triangle whose vertices are u_{h-1}, v_{h-1}, and w_{h-1}; the catheti (u_{h-1}, w_{h-1}) and (v_{h-1}, w_{h-1}) lie on lines parallel to the bisectors of the axes, so that u_{h-1} and w_{h-1} lie on the same line l_{uw} with slope $\pi/4$, with u_{h-1} below and to the left of w_{h-1}, and so that v_{h-1} and w_{h-1} lie on the same line l_{vw} with slope $3\pi/4$, with v_{h-1} below and to the right of w_{h-1}; the hypothenuse lies on an horizontal line l_{vw}.

Denote by a the line obtained by translating l_{vw} by $(1,0)$. If h is even, shift vertex u_{h-1} by $(-1,-1)$ and vertex v_{h-1} by $(1,1)$. If h is odd, shift vertex u_{h-1} by $(-1,0)$ and vertex v_{h-1} by $(1,0)$. Now, reflect the modified drawing Γ_{h-1} with respect to a. Insert the edge from u_{h-1} to its symmetric vertex, say z. Let $u_h = u_{h-1}$, $v_h = z$ and $w_h = v_{h-1}$ (see Fig. 5.1).

It's easy to see that the inductive construction of the algorithm builds drawings satisfying the hypotheses. Examples of the drawings produced by the algorithm are shown in Fig. 5.2.

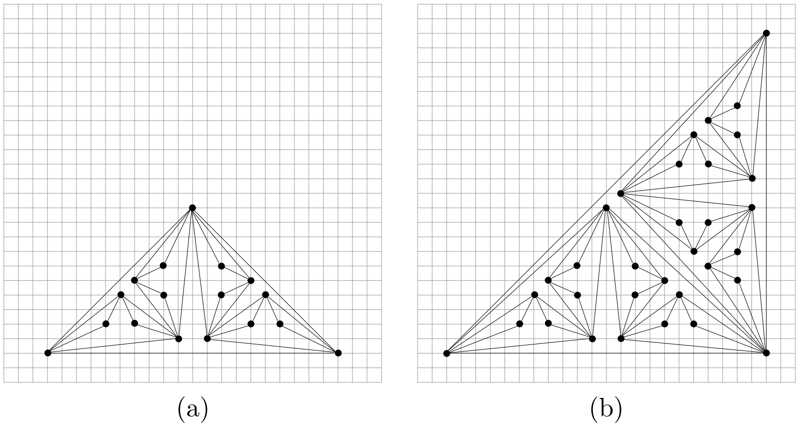
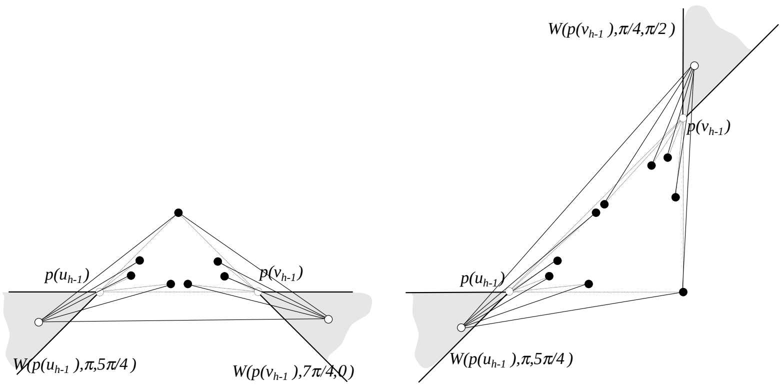


Figure 5.2: Applications of the algorithm in *Sect. 5.2*. (a) Γ_4. (b) Γ_5.

We prove the planarity of such drawings.

Lemma 5.1 Γ_h *is planar.*

Proof: Suppose, by induction, that the drawing Γ_{h-1} of G_{h-1} constructed by the algorithm has the following properties: **P**roperty (i) Γ_{h-1} is planar. **P**roperty (ii) denote by $p(u_{h-1})$ and $p(v_{h-1})$ the points where u_{h-1} and v_{h-1} are drawn in Γ_{h-1}, respectively; if h is even it is possible to simultaneously shift vertex u_{h-1} to any point of the closed wedge $W(p(u_{h-1}), \pi, 5\pi/4)$ centered at $p(u_{h-1})$ and delimited by the half-lines with slopes π and $5\pi/4$ and vertex v_{h-1} to any point of the closed wedge $W(p(v_{h-1}), \pi/4, \pi/2)$ centered at $p(v_{h-1})$ and delimited by the half-lines with slopes $\pi/4$ and $\pi/2$, respectively, without altering the planarity of Γ_{h-1} (see Fig. 5.3 (a)); if h is odd it is possible to simultaneously shift vertex u_{h-1} to any point of the closed wedge $W(p(u_{h-1}), \pi, 5\pi/4)$ centered at $p(u_{h-1})$ and delimited by the half-lines with slopes π and $5\pi/4$ and vertex v_{h-1} to any point of the closed wedge $W(p(v_{h-1}), 7\pi/4, 0)$ centered at $p(v_{h-1})$ and delimited by the half-lines with slopes $7\pi/4$ and 0, respectively, without altering the planarity of Γ_{h-1} (see Fig. 5.3 (b)).



(a) (b)

Figure 5.3: Wedges in the proof of planarity of Γ_h. (a) h odd. (b) h even.

Note that properties (i) and (ii) are trivially satisfied by Γ_1. Inductively suppose that properties (i) and (ii) are satisfied by Γ_{h-1}. First, suppose h is even.

We show that Γ_h satisfies property (i). After vertex u_{h-1} has been shifted by $(-1, -1)$ and vertex v_{h-1} by $(1, 1)$, such vertices are still inside wedges $W(p(u_{h-1}), \pi, 5\pi/4)$ and $W(p(v_{h-1}), \pi/4, \pi/2)$, respectively. Hence, by property (ii) of Γ_{h-1}, the planarity of Γ_{h-1} is maintained by such shifts. The reflection of the modified Γ_{h-1} with respect to line a doesn't cause crossings, since a is tangent to the modified Γ_{h-1}. Since vertices u_h and v_h are below any other vertex of G_h in the resulting drawing, then inserting edge (u_h, v_h) doesn't create crossing. Hence, Γ_h is planar.

Now we show that Γ_h satisfies property (ii). Denote by Γ^1_{h-1} the sub-drawing of Γ_h induced by the vertices to the left of a and by vertex w_h; denote also by Γ^2_{h-1} the sub-drawing of Γ_h induced by the vertices to the right of a and by vertex w_h. Denote by $p(u_h)$ and by $p(v_h)$ the points where u_h and v_h are drawn in Γ_h, respectively. Denote by $p^{-1}(u_h)$ (by $p^{-1}(v_h)$) the point one unit to the right and one unit above $p(u_h)$ (the point one unit to the left and one unit above $p(v_h)$). Since $p(u_h)$ is in the closed wedge $p^{-1}(u_h), \pi, 5\pi/4)$, then $W(p(u_h), \pi, 5\pi/4)$ is entirely contained in $W(p^{-1}(u_h)), \pi, 5\pi/4)$ (part of their delimiting half-lines coincide). Symmetrically, $W(p(v_h), 7\pi/4, 0)$ is in the closed wedge $W(p^{-1}(v_h), 7\pi/4, 0)$. Suppose that property (ii) is not satisfied by Γ_h. Shifting u_h inside $W(p(u_h), \pi, 5\pi/4)$ and v_h inside $W(p(v_h), 7\pi/4, 0)$ doesn't create a crossing between any edge $e_1 \in \Gamma^1_{h-1}$ and any edge $e_2 \in \Gamma^2_{h-1}$, since e_1 and e_2 are separated by a. Shifting u_h inside $W(p(u_h), \pi, 5\pi/4)$ doesn't create a crossing between any edge $e_a \in \Gamma^1_{h-1}$ and any edge $e_b \in \Gamma^1_{h-1}$, or between any edge $e_a \in \Gamma^2_{h-1}$ and any edge $e_b \in \Gamma^2_{h-1}$, since such a crossing would imply that property (i) is not verified by Γ_{h-1}. Finally, shifting u_h inside $W(p(u_h), \pi, 5\pi/4)$ and v_h inside $W(p(v_h), 7\pi/4, 0)$ doesn't create a crossing between edge (u_h, v_h) and any other edge, since regardless of the position of u_h and v_h inside the corresponding wedges, such vertices are below any other vertex of the drawing.

If h is odd, properties (i) and (ii) are inductively verified analogously. □

Now we analyze the area requirement of the drawings obtained by applying the above described algorithm.

Lemma 5.2 *If h is even, then the height of Γ_h is $\frac{7}{2}\sqrt{n-1}-3$ and its width is $7\sqrt{n-1}-7$. If h is odd, then the height of Γ_h is $\frac{7}{\sqrt{2}}\sqrt{n-1}-5$ and its width is $\frac{7}{\sqrt{2}}\sqrt{n-1}-5$.*

Proof: Let $height_h$ and $width_h$ be the height and the width of Γ_h, respectively.

- ***h is even:*** $height_h$ is given by the vertical distance between u_h and w_h, that is equal to the vertical distance between u_{h-1} and v_{h-1} in Γ_{h-1} after their shifts, that is (1) $height_h = height_{h-1} + 2$. Further, $width_h$ is given by the horizontal distance between u_h and v_h. Such a distance is twice the horizontal distance between u_{h-1} and v_{h-1} in Γ_{h-1} after their shifts, minus one unit corresponding to the vertical grid line a, that is counted twice. Hence, we have (2) $width_h = 2\cdot(width_{h-1}+2)-1 = 2\cdot width_{h-1}+3$.

- ***h is odd:*** $height_h$ is given by the vertical distance between v_h and w_h, that is equal to the horizontal distance between u_{h-1} and v_{h-1} in Γ_{h-1} after their shifts, that is (3) $height_h = width_{h-1} + 2$. Further, $width_h$ is also given by the horizontal distance between u_{h-1} and v_{h-1} in Γ_{h-1} after their shifts. Hence, we have (4) $width_h = width_{h-1} + 2$.

Let us compute $width_h$ as a function of h and, consequently, as a function of n. Suppose h is even. Substituting (4) into (2) we get $width_h = 2\cdot width_{h-2}+7$. Using $width_2 = 7$, a simple inductive proof shows that (5) $width_h = 7 \cdot 2^{\frac{h}{2}} - 7$, which is $7\sqrt{n-1} - 7$, since $h = \log_2(n-1)$. If h is odd, we derive the value of $width_h$ by substituting (5) into (4). Namely, (6) $width_h = 7 \cdot 2^{\frac{h-1}{2}} - 7 + 2 = \frac{7}{\sqrt{2}} \cdot 2^{\frac{h}{2}} - 5$. Hence, $width_h = \frac{7}{\sqrt{2}}\sqrt{n-1} - 5$.

Now we compute $height_h$. Suppose h is odd. Substituting (5) into (3), we get (7) $height_h = 7 \cdot 2^{\frac{h-1}{2}} - 7 + 2 = \frac{7}{\sqrt{2}} \cdot 2^{\frac{h}{2}} - 5$ and hence $height_h = \frac{7}{\sqrt{2}}\sqrt{n-1} - 5$. Finally, suppose h is even. Substituting (7) into (1), we get (8) $height_h = \frac{7}{\sqrt{2}} \cdot 2^{\frac{h-1}{2}} - 5 + 2 = \frac{7}{2} \cdot 2^{\frac{h}{2}} - 3$ and hence $height_h = \frac{7}{2}\sqrt{n-1} - 3$. □

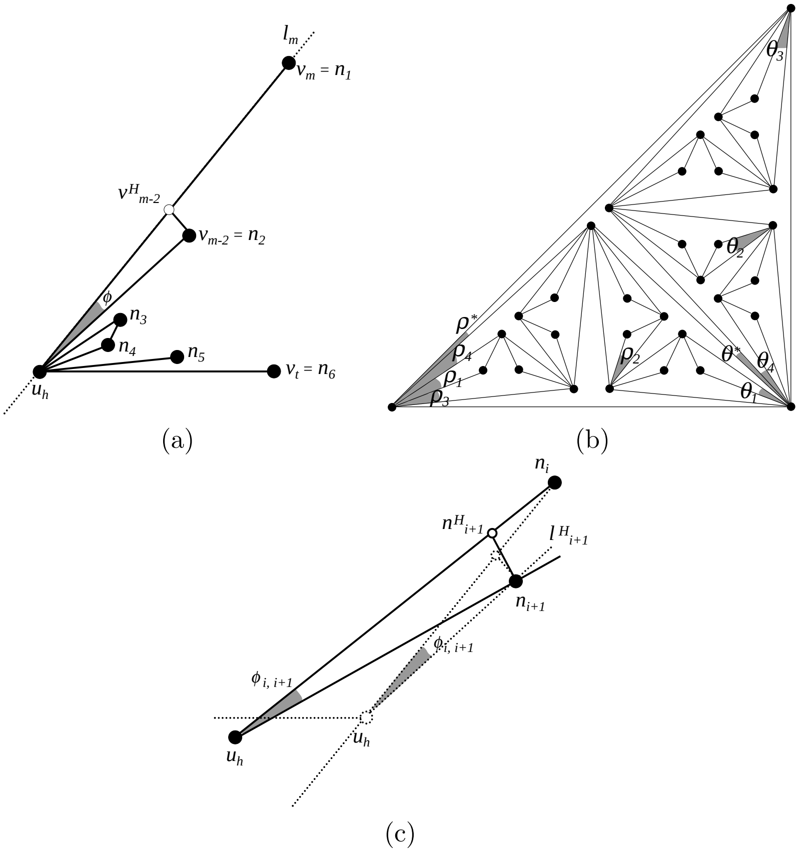
We analyze the angular resolution of the drawings built by the above algorithm. Let u_h be the left vertex of Γ_h. Notice that $u_h = u_{h-1} = \ldots = u_1$. Constructing Γ_h from Γ_{h-1}, the number of neighbors of u_h increases by one. Let v_i be the neighbor of u_h inserted in the construction of Γ_i. Let m (resp. t) be the largest odd (resp. even) integer less or equal than h.

Let ϕ be the angle between segments $\overline{u_h v_{m-2}}$ and $\overline{u_h v_m}$ (see Fig. 5.4 (a)). Denote by v^H_{m-2} the intersection point between the line l_m through u_h and v_m and the line through v_{m-2} orthogonal to l_m.

Lemma 5.3 $\sin\phi > k/\sqrt{n}$, *for some constant k.*

Proof: Observe that, independently of the value of m, the length $|\overline{v^H_{m-2} v_{m-2}}|$ of segment $\overline{v^H_{m-2} v_{m-2}}$ is $\sqrt{2}/2$. Since Γ_h is contained in a bounding-box of $O(\sqrt{n}) \times O(\sqrt{n})$, the length $|\overline{u_h v_{m-2}}|$ of segment $\overline{u_h v_{m-2}}$ is $O(\sqrt{n})$ and hence $\sin\phi = |\overline{v^H_{m-2} v_{m-2}}|/|\overline{u_h v_{m-2}}| > k/\sqrt{n}$, for some constant k. □

Lemma 5.4 *ϕ is the smallest angle of Γ_h.*



(a)

(b)

(c)

Figure 5.4: Illustration for the proof of the bound on the angular resolution of Γ_h.

Proof: We prove the lemma by induction on h. As base case of the induction, consider Γ_1. Trivially, $\phi = \pi/4$ is the smallest angle of the drawing. Now suppose the claim holds for Γ_{h-1}. Denote by Γ^S_{h-1} the drawing obtained from Γ_{h-1} after shifting vertices u_{h-1} and v_{h-1}. The first part of the proof consists of showing that it is sufficient to compare ϕ only with the angles incident to u_h. The second part of the proof shows that ϕ is the smallest of such angles.

Angle ϕ is less in Γ_h than in Γ_{h-1}. Namely, $|\overline{v^H_{m-2}v_{m-2}}|$ is still $\sqrt{2}/2$ and the length $|\overline{u_h v_{m-2}}|$ of segment $\overline{u_h v_{m-2}}$ is larger in Γ_h than in Γ_{h-1}, hence $\sin\phi = |\overline{v^H_{m-2} v_{m-2}}|/|\overline{u_h v_{m-2}}|$ is less in Γ_h than in Γ_{h-1}. Hence, for every angle θ in Γ^S_{h-1} that is not incident to u_{h-1} or to v_{h-1} we have $\theta > \phi$ by inductive hypothesis. It follows that, to determine the smallest angle in Γ_h, we can forget about any angle also belonging to Γ^S_{h-1} and not incident to u_{h-1} or v_{h-1}.

Observe that (see Fig. 5.4 (b)): (i) for every angle θ_1 incident to v_{h-1} in Γ^S_{h-1} there is an angle ρ_1 incident to u_{h-1} in Γ^S_{h-1} such that $\rho_1 = \theta_1$, (ii) for every angle θ_2 not incident to u_h, v_h, and w_h in Γ_h, there is an angle ρ_2 in Γ^S_{h-1} such that $\rho_2 = \theta_2$, (iii) for every angle θ_3 incident to v_h in Γ_h there is an angle ρ_3 incident to u_h in Γ_h such that $\rho_3 = \theta_3$, (iv) for every angle θ_4 incident to w_h in Γ_h, but for one angle θ^*, there is an angle ρ_4 in Γ^S_{h-1} such that $\rho_4 = \theta_4$, and (v) angle θ^* is the angle between segments $\overline{w_h w_{h-1}}$ and $\overline{w_h w^*_{h-1}}$, where w^*_{h-1} is the vertex obtained by reflecting w_{h-1} with respect to a, and there exists an angle ρ^* incident to u_h in Γ_h such that $\theta^* = 2 \cdot \rho^*$. Such observations can be trivially proved exploiting the symmetry of the drawings constructed by the algorithm. By the inductive hypothesis it follows that, to determine the smallest angle in Γ_h, we can look only at angles incident to u_h.

Consider the neighbors of u_h and consider the clockwise order ($t_1 = v_m, t_2, \dots, t_h, t_{h+1} = v_t$) in which they are incident to u_h. Observe that $t_{\lceil h/2 \rceil}$ and $t_{\lceil h/2 \rceil + 1}$ are adjacent. Denote by $\phi_{i,i+1}$ the angle incident to u_h delimited by segments $\overline{u_h t_i}$ and $\overline{u_h t_{i+1}}$, with $i = 1, 2, \dots, h$. We have $\phi = \phi_{1,2}$ and we claim that such an angle is the smallest among all angles $\phi_{i,i+1}$, with $i = 1, 2, \dots, h$.

First, we show that $\sin\phi_{i,i+1} > \sin\phi_{1,2}$, for $i = 2, 3, \dots, \lceil h/2 \rceil$. Namely, denoting by t^H_{i+1} the intersection point between the line l^H_i through u_h and t_i, and the line through t_{i+1} orthogonal to l^H_i, we have $\sin\phi_{i,i+1} = |\overline{t^H_{i+1} t_{i+1}}|/|\overline{u_h t_{i+1}}|$. We prove that $|\overline{t^H_{i+1} t_{i+1}}| > \sqrt{2}/2$, for any $i = 2, 3, \dots, \lceil h/2 \rceil$. Consider the placement of the left vertex u_h in the drawing constructed at the step of the algorithm when vertex t_i was first introduced. In such a drawing, the slope of the line through u_h and t_i was $\pi/4$ by construction and $|\overline{t^H_{i+1} t_{i+1}}| = \sqrt{2}/2$. In the inductive steps that lead to the construction of Γ_h, vertices t_i and t_{i+1} do not move and vertex u_h moves in turn by $(-1, 0)$ and by $(-1, -1)$. Hence, the slope of the line through u_h and t_i is less than $\pi/4$ and $\overline{t^H_{i+1} t_{i+1}} > \sqrt{2}/2$ (see Fig. 5.4 (c)). It is trivial to prove that $\overline{u_h t_2}$ is the longest among segments $\overline{u_h t_{i+1}}$, with $i = 1, 2, \dots, \lceil h/2 \rceil$. It follows that $\sin\phi_{i,i+1} > \sin\phi$, for any $i = 2, 3, \dots, \lceil h/2 \rceil$.

Next, we show that $\sin\phi_{i,i+1} > \sin\phi_{h,h+1}$, for any $i = \lceil h/2 \rceil + 1, \lceil h/2 \rceil +$

$2, \ldots, h-1$. Denoting by t_i^H the intersection point between the line l_{i+1}^H through u_h and t_{i+1}, and the line through t_i orthogonal to l_{i+1}^H, we have $\sin\phi_{i,i+1} = |\overline{t_i^H t_i}|/|\overline{u_h t_i}|$. Analogously to the previous case, it can be proved that $|\overline{t_i^H t_i}| > 1$, for any $i = \lceil h/2 \rceil + 1, \lceil h/2 \rceil + 2, \ldots, h-1$, that $|\overline{t_h^H t_h}| = 1$, and that $\overline{u_h t_h}$ is the longest among segments $\overline{u_h t_i}$, with $i = \lceil h/2 \rceil + 1, \lceil h/2 \rceil + 2, \ldots, h$. It follows that $\sin\phi_{i,i+1} > \sin\phi_{h,h+1}$, for any $i = \lceil h/2 \rceil + 1, \lceil h/2 \rceil + 2, \ldots, h-1$.

It remains just to compare $\sin\phi_{1,2}$ with $\sin\phi_{h,h+1}$. Such values can be easily derived from the results on the width and the height of Γ_h, by applying the Pythagorean theorem to compute $\overline{u_h t_2}$ and $\overline{u_h t_h}$.

Namely, if h is odd we have: $\sin\phi_{1,2} = \frac{\sqrt{2}}{(2 \cdot \sqrt{(7 \cdot 2^{(h-3/2)} - 3)^2 + (7 \cdot 2^{(h-3/2)} - 4)^2})} < \frac{1}{\sqrt{(7 \cdot 2^{(h-3/2)} - 4)^2 + 1}} = \sin\phi_{h,h+1}$. If h is even, we have: $\sin\phi_{1,2} = \frac{\sqrt{2}}{(2 \cdot \sqrt{(7 \cdot 2^{(h-4/2)} - 2)^2 + (7 \cdot 2^{(h-4/2)} - 3)^2})} = \frac{1}{\sqrt{\frac{49}{4} \cdot 2^h - 35 \cdot 2^{h/2} + 26}}$. Also, we have $\sin\phi_{h,h+1} = 1/\sqrt{(7 \cdot 2^{(h-2/2)} - 5)^2 + 1} = 1/\sqrt{\frac{49}{4} \cdot 2^h - 35 \cdot 2^{h/2} + 26} = \sin\phi_{1,2}$. Hence, ϕ is the smallest angle in Γ_h. □

From the above discussion we have:

Theorem 5.1 *Given an n-vertex complete outerplanar graph G_h with height h, there exists an $O(n)$-time algorithm that constructs a planar straight-line grid drawing Γ_h of G_h such that:*

1. *if h is even, then the height of Γ_h is $\frac{7}{2}\sqrt{n-1} - 3$ and its width is $7\sqrt{n-1} - 7$;*

2. *if h is odd, then the height of Γ_h is $\frac{7}{\sqrt{2}}\sqrt{n-1} - 5$ and its width is $\frac{7}{\sqrt{2}}\sqrt{n-1} - 5$;*

3. *the angular resolution of Γ_h is greater than $\frac{k}{\sqrt{n}}$, with k constant;*

4. *isomorphic subgraphs of G_h have congruent drawings in Γ_h up to a translation and a reflection; and*

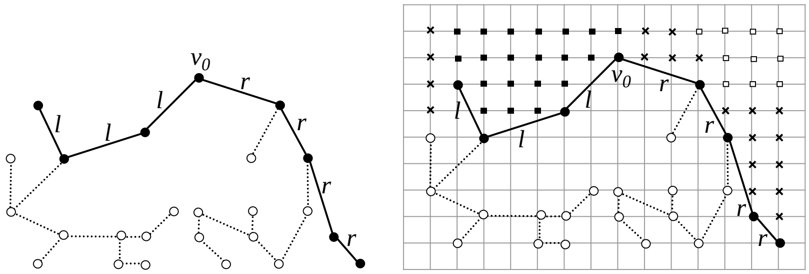
5. *Γ_h is axially symmetric.*

From the fact that a balanced outerplanar graph can be augmented to complete by adding a linear number of nodes and without altering its height we have:

Theorem 5.2 *Given an n-vertex balanced outerplanar graph G, there exists an $O(n)$-time algorithm that constructs a planar straight-line grid drawing Γ of G such that both the height and the width of Γ are $O(\sqrt{n})$ and the angular resolution of Γ is greater than $\frac{k}{\sqrt{n}}$, with k constant.*

5.3 Outerplanar Drawings and Star-Shaped Drawings

In this section we show that the planar drawings of outerplanar graphs are strongly related to the drawings of their dual trees.



(a) (b)

Figure 5.5: A binary tree T. The edges (u, v) labeled r (l) are such that v is the right (resp. left) child of u. (a) The thick edges and the black vertices show $L(T)$ and $R(T)$. (b) A straight-line grid drawing of T. The squared black points and the squared white points are examples of points of the outer-left set and of the outer-right set, respectively. The 'x' points don't belong to either one of the sets.

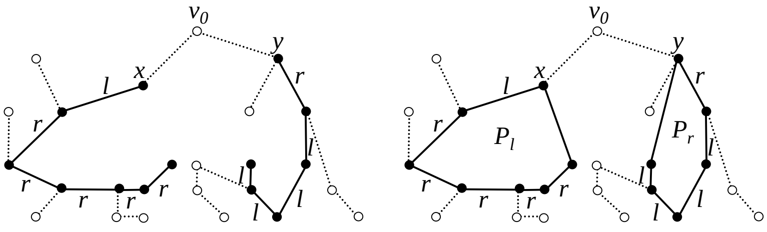
Let T be a binary tree rooted at any node v_0. The *leftmost path* $L(T)$ (the *rightmost* path $R(T)$) of T is the path $v_0, v_1, \ldots, v_m$ such that v_{i+1} is the left child (resp. the right child) of v_i, $\forall i$ such that $0 \leq i \leq m-1$, and v_m doesn't have a left child (resp. a right child) (see Fig. 5.5 (a)).

The *outer-left set* (*outer-right set*) of a planar straight-line drawing Γ of T is the set of points of the plane with integer coordinates from which it's possible to draw edges to each one of the nodes of $L(T)$ (resp. of $R(T)$) without creating crossings with the edges of Γ (see Fig. 5.5 (b)).

The *left-right* (*right-left*) *path* of a node $t \in T$ is the path $v_0, v_1, \ldots, v_m$ such that $v_0 = t$, v_1 is the left child (resp. the right child) of v_0, v_{i+1} is the right child (resp. the left child) of v_i, $\forall i$ such that $1 \leq i \leq m-1$, and v_m doesn't have a right child (resp. a left child) (see Fig. 5.6 (a)).

The *left cycle of the neighbors* (*right cycle of the neighbors*) of a node $t \in T$, with left-right path $v_0 = t, v_1, \dots, v_m$ (with right-left path $u_0 = t, u_1, \dots, u_p$), is defined as follows: if $m = 0$ (if $p = 0$), then the left cycle of the neighbors (the right cycle of the neighbors) is vertex t; if $m = 1$ (if $p = 1$), then the left cycle of the neighbors (the right cycle of the neighbors) is edge (v_0, v_1) (is edge (u_0, u_1)); if $m \geq 2$ (if $p \geq 2$), then the left cycle of the neighbors (the right cycle of the neighbors) is the cycle composed of the edges of the left-right path of t (resp. of the edges of the right-left path of t) plus an extra edge connecting v_m and v_0 (resp. plus an extra edge connecting u_p and u_0). See Fig. 5.6 (b).

Consider any drawing Γ of T. The *left polygon of the neighbors* $P_l(t)$ (resp. the *right polygon of the neighbors* $P_r(t)$) of a node $t \in T$ is the polygon of representing in Γ the left cycle of the neighbors (resp. the right cycle of the neighbors).



(a) (b)

Figure 5.6: A binary tree T. The edges (u, v) labeled r (l) are such that v is the right (resp. left) child of u. (a) The thick edges and the black vertices show the left-right path of node x and the right-left path of node y in T. (b) The thick edges, the black vertices and the labels P_r and P_l show the left cycle of the neighbors of node x and the right cycle of the neighbors of node y, respectively.

A planar straight-line order-preserving drawing Γ of T is *star-shaped* if it satisfies all the following conditions:

1. For each node $t \in T$, the left (right) polygon of the neighbors $P_l(t)$ (resp. $P_r(t)$) representing the left cycle of the neighbors $(t, v_1, \dots, v_m)$ (resp. the right cycle of the neighbors $(t, u_1, \dots, u_p)$), is a simple polygon; further, if $m > 2$ (if $p > 2$) each segment (t, v_i), with $2 \leq i \leq m-1$ (resp. each

segment (t, u_j), with $2 \leq j \leq p-1$) belongs to the interior of $P_l(t)$ (resp. of $P_r(t)$), but for its endpoints t and v_i.

2. For each node $t \in T$, $P_l(t)$ ($P_r(t)$) does not contain any vertex of T in its interior and does not contain any vertex of T on its border, but for the vertices of the left (resp. right) cycle of the neighbors of t.

3. There exist a point p_u in the outer-left set of Γ and a point p_v in the outer-right set of Γ such that segment (p_u, p_v) doesn't intersect any edge of Γ.

Observe that the drawings of trees in Figs. 5.5 and 5.6 are star-shaped.

Given an outerplanar graph G and its poles u and v, we call *internal subgraph* $I(G)$ the graph obtained by deleting u, v and their incident edges from G.

We now put in evidence a tight relation between the drawing of an n-vertex maximal outerplanar graph G and the drawing of its dual binary tree T. First, we establish a bijection γ between the nodes of T and the vertices of G, but for its poles (see Fig. 5.7).

Consider the poles u and v and the central vertex w of G. By definition such vertices are incident to an internal face f of an outerplanar embedding $\mathcal{G}$ of G. Further, edge (u, v) is incident to the outer face of $\mathcal{G}$. Face f corresponds to the root v_0 of T. Set $\gamma(v_0) = w$. Label also edge (u, w) as a *left edge* and edge (v, w) as a *right edge.* A left (right) edge is such that its dual edge connects a node to its left (resp. right) child in T.

Consider the order of the vertices of G given by its construction as a 2-tree, i.e., the order $O : (u, v, w, v_1, v_2, \ldots, v_{n-3})$ of the vertices of G starting from u, v, and w, and such that v_1 is adjacent to exactly two out of u, v, and w, and, for $i = 2, \ldots, n-3$, exactly two adjacent vertices u_i^1 and u_i^2 of $(u, v, w, v_1, v_2, \ldots, v_{i-1})$ are neighbors of v_i.

Denote by G_0 the subgraph of G induced by u, v, and w. Denote by G_i the subgraph of G induced by vertices $(u, v, w, v_1, v_2, \ldots, v_i)$, for $i = 1, \ldots, n-3$. From the construction of an outerplanar graph as a 2-tree, we have:

Property 5.1 *For each $i = 0, 1, \ldots, n-3$, G_i is a maximal outerplanar graph.*

We construct the outerplanar embedding $\mathcal{G}$ of G starting from an embedding $\mathcal{G}_0$ of G_0 and by adding vertices v_i's in the order O defined above. When vertex v_i is inserted together with edges (v_i, u_i^1) and (v_i, u_i^2), partial embedding $\mathcal{G}_i$ is constructed.

When $\mathcal{G}_i$ is constructed by the addition of vertex v_i, each vertex $v_j \in (w, v_1, v_2, \ldots, v_{i-1})$ has been already put in bijection with an internal face f_j of $\mathcal{G}_i$ that is incident to v_j. Inductively assume that each edge $e \neq (u, v)$ incident to the outer face of $\mathcal{G}_i$ is such that the internal face of $\mathcal{G}_i$ incident to e has been put in bijection with one of the end-vertices of e.

The insertion of vertex v_i and of its incident edges (v_i, u_i^1) and (v_i, u_i^2) creates a new internal face f_i with incident vertices v_i, u_i^1, and u_i^2. Consider node $t(f_i)$ of the dual tree of $\mathcal{G}_i$ corresponding to f_i and the node $p(f_i)$ of the dual tree of $\mathcal{G}_i$ parent of $t(f_i)$. By induction, the face of $\mathcal{G}_i$ corresponding to $p(f_i)$ has been mapped with one out of u_i^1 and u_i^2. Suppose, w.l.o.g., that $\gamma(p(f_i)) = u_i^1$. Set $\gamma(t(f_i)) = v_i$, satisfying the inductive hypothesis for both edges (v_i, u_i^1) and (v_i, u_i^2) that are incident to the outer face. If edge (u_i^1, u_i^2) was a left edge then label (v_i, u_i^1) as a right edge and (v_i, u_i^2) as a left edge. If edge (u_i^1, u_i^2) was a right edge then label (v_i, u_i^1) as a left edge and (v_i, u_i^2) as a right edge.

Denote by $f(t)$ the face of $\mathcal{G}$ corresponding to the node t of T.

Property 5.2 *Each node t of T is associated to the vertex incident to $f(t)$ that comes last in O.*

The following lemma proves the interesting property that, given any maximal outerplanar graph G, it dual tree T is a subgraph of $I(G)$ and hence of G itself.

Lemma 5.5 *If edge (t_i, t_j) belongs to T, then edge $(\gamma(t_i), \gamma(t_j))$ belongs to G.*

Proof: Suppose w.l.o.g. that t_i is the parent of t_j in T. Let $v_i = \gamma(t_i)$ and $v_j = \gamma(t_j)$. Consider the order $O : (u, v, w, v_1, v_2, \ldots, v_{n-3})$ of the vertices of G defined above. Since t_i is the parent of t_j in T, then v_j follows v_i in O. Denote by u_j^1 and by u_j^2 the neighbors of v_j in G_{j-1}. Hence, faces $f(t_i)$ and $f(t_j)$ share edge (u_j^1, u_j^2) that is dual to edge (t_i, t_j) of T. Further, by the inductive construction of the bijection between the vertices of G and the nodes of T, node t_i is mapped to one out of u_j^1 and by u_j^2. Since G contains both edges (v_j, u_j^1) and (v_j, u_j^2), the statement follows. □

Let G be any maximal outerplanar graph and T its dual binary tree. Let $I(G)$ be the internal subgraph of G. Denote by $(t, v_1, \ldots, v_m)$ (by $(t, u_1, \ldots, u_p)$) the left cycle of the neighbors (the right cycle of the neighbors) of node $t \in T$. For each node $t \in T$, if $m > 1$ add to T edges connecting t to each node v_i, with $2 \leq i \leq m$, and, if $p > 1$ add to T edges connecting t to each node u_i, with $2 \leq i \leq p$. Denote by T^+ the resulting graph.

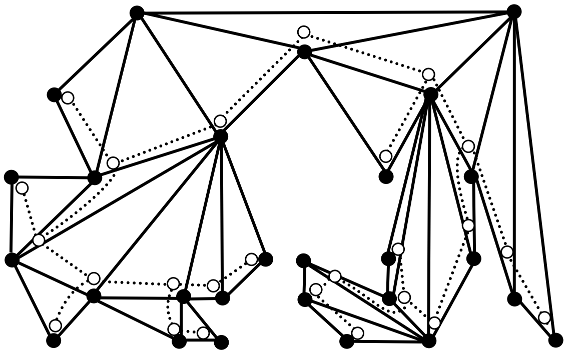


Figure 5.7: Bijection between the small white nodes of the dual tree T and the big black vertices of the primal outerplanar graph G.

Lemma 5.6 $T^+ = I(G)$.

Proof: Consider the order $O : (u, v, w, v_1, v_2, \dots, v_{n-3})$ of the vertices of G in its construction as a 2-tree. As before, denote by G_0 the outerplanar graph induced by vertices u, v, w, by G_i the outerplanar graph induced by vertices $u, v, w, v_1, v_2, \dots, v_i$, and denote also by T_i the dual binary tree of G_i. We prove by induction on i that T_i^+ is the internal subgraph of G_i.

In the base case, graph T_0^+ consists only of node $\gamma^{-1}(w)$ and the internal subgraph of G_0 consists only of vertex w. Inductively suppose that T_{i-1}^+ is the internal subgraph of G_{i-1}.

Consider the outerplanar embedding $\mathcal{G}_i$ of G_i. Let f_i be the internal face created by the insertion of v_i and its incident edges (v_i, u_i^1) and (v_i, u_i^2) in $\mathcal{G}_{i-1}$. Let $t(f_i)$ be the node of T corresponding to f_i and let $p(f_i)$ be the parent of $t(f_i)$ in T. We know that $\gamma(t(f_i)) = v_i$. Further, we can assume w.l.o.g. that $\gamma(p(f_i)) = u_i^1$.

Suppose that $t(f_i)$ is the left child of $p(f_i)$. Consider the path $(t_1, t_2, \dots, t_{k-2}, t_{k-1} = p(f_i), t_k = t(f_i))$ such that: (i) for any $j = 2, 3, \dots, k-1$, node t_{j+1} is the left child of node t_j; (ii) t_2 is the right child of t_1 or t_1 is the root of T. Notice that such a path always exists and is unique.

Distinguish two cases: (1) node t_2 is the left child of t_1; (2) node t_2 is the right child of t_1.

In the first case t_1 is the root of T (see Fig. 5.8 (a)). By definition of poles, we have $u_i^2 = u$, that is, u_i^2 is the left vertex of G. Hence, the internal subgraph

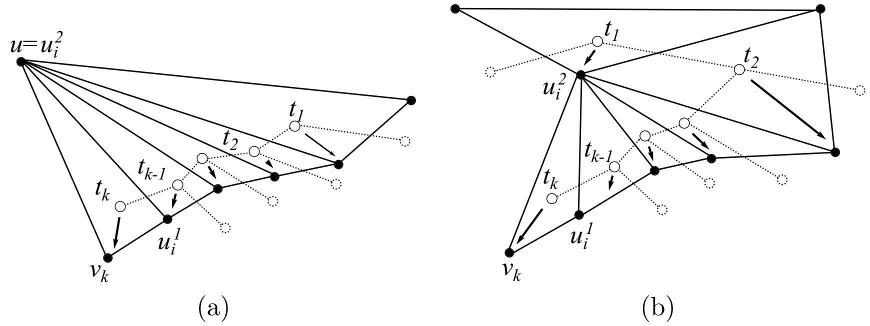


Figure 5.8: Illustration for the proof of Lemma 5.6

of G_i differs from the internal subgraph of G_{i-1} only for vertex v_i and edge (v_i, u_i^1). We show that T_i^+ contains the internal subgraph of G_{i-1}, edge (v_i, u_i^1), and no other edge. Graph T_i^+ contains T_{i-1}^+ and hence, by inductive hypothesis, the internal subgraph of G_{i-1}. Further, by Lemma 5.5 it contains edge (v_i, u_i^1) since $\gamma^{-1}(v_i) = t(f_i)$, since $\gamma^{-1}(u_i^1) = p(f_i)$, and since $(t(f_i), p(f_i)) \in T$. Node $t(f_i)$ does not belong to the right cycle of the neighbors of any node of T_i. Further, $t(f_i)$ belongs to the left cycle of the neighbors of node $p(f_i)$ and does not belong to the left cycle of the neighbors of any other node. However, since $t(f_i)$ is the second node in the left cycle of the neighbor of $p(f_i)$, no further edge incident to $t(f_i)$ is added in T_i^+.

In the second case the internal subgraph of G_i differs from the internal subgraph of G_{i-1} for vertex v_i and edges (v_i, u_i^1) and (v_i, u_i^2) (see Fig. 5.8 (b)). We show that T_i^+ contains the internal subgraph of G_{i-1}, edges (v_i, u_i^1) and (v_i, u_i^2), and no other edge. Graph T_i^+ contains T_{i-1}^+ and hence, by inductive hypothesis, the internal subgraph of G_{i-1}. Further, by Lemma 5.5 it contains edge (v_i, u_i^1). Node $t(f_i)$ belongs to the left cycle of the neighbors of node $p(f_i)$ and does not belong to the left cycle of the neighbors of any other node. However, since $t(f_i)$ is the second node in the left cycle of the neighbor of $p(f_i)$, no further edge incident to $t(f_i)$ is added in T_i^+. Node $t(f_i)$ belongs to the right cycle of the neighbors of t_1 and does not belong to the right cycle of the neighbors of any other node. Hence, edge $(t_1, t(f_i))$ is in T_i^+. Since $\gamma(t(f_i)) = v_i$, it remains only to show that $\gamma(t_1) = u_i^2$. Consider edge (u_i^2, u_i^3) dual to edge (t_1, t_2). By inductive hypothesis in the construction of the bijection, t_1 has been mapped to one out of u_i^2 and u_i^3. However, if it is $\gamma(t_1) = u_i^3$, then (t_1, t_2) would be a left edge.

The case in which $t(f_i)$ is the right child of $p(f_i)$ can be discussed analogously. □

Conversely, let G be a maximal outerplanar graph with dual tree T. Let $I(G)$ be the internal subgraph of G. For each vertex v_i of $I(G)$, let t be the node of T such that $\gamma^{-1}(v_i) = t$, let p, t_l, and t_r be the parent, the left child, and the right child of t in T, respectively. Remove from $I(G)$ all the edges incident on v_i, but for $(v_i, \gamma(p))$, $(v_i, \gamma(t_l))$, and $(v_i, \gamma(t_r))$. Denote by $I^-(G)$ the resulting graph.

Lemma 5.7 *$I^-(G) = T$.*

Proof: By construction, all the edges $(v_i, v_j) \in I^-(G)$ are such that $(\gamma^{-1}(v_i), \gamma^{-1}(v_j)) \in T$. Further, suppose that an edge $(t_i, t_j) \in T$ is not in I^-. Assume, w.l.o.g., that t_i is parent of t_j. Consider vertices $v_i = \gamma(t_i)$ and $v_j = \gamma(t_j)$ in G. By Lemma 5.5, edge (v_i, v_j) belongs to G and it could not have been removed from $I(G)$ since, by construction, edges connecting vertices of $I(G)$ whose corresponding nodes are neighbors in T are not removed. □

Lemma 5.8 *Let G be an n-vertex outerplanar graph and suppose that its dual tree T admits a star-shaped drawing with $f(n)$ area. We have that G admits an outerplanar straight-line drawing such that the area of the drawing of its internal subgraph is $f(n)$.*

Proof: Suppose you have a planar star-shaped straight-line grid drawing Γ of T. Map each vertex v_i of G, but for its poles, to the point where the node t such that $\gamma^{-1}(v_i) = t$ is drawn.

For each node $t \in T$ with left cycle of the neighbors $(t, v_1, \dots, v_m)$ (with right cycle of the neighbors $(t, u_1, \dots, u_p)$), if $m > 1$ draw segments connecting t to each node v_i, $2 \leq i \leq m$, and, if $p > 1$ draw segments connecting t to each node u_i, $2 \leq i \leq p$. By Lemma 5.6 the resulting graph $I(G)$ is the internal subgraph of G. Denote by $\Gamma_{I(G)}$ the resulting drawing of $I(G)$. By Conditions 1 and 2 of a star-shaped drawing, we have that $\Gamma_{I(G)}$ is a planar drawing. Since $\Gamma_{I(G)}$ is straight-line, and its vertices have the same coordinates of the nodes in Γ, then the area of $\Gamma_{I(G)}$ is equal to the area of Γ.

By definition of internal subgraph, it is sufficient to insert the poles of G and to draw the edges between the poles and the vertices of $L(T)$ and $R(T)$ to augment $\Gamma_{I(G)}$ in a straight-line outerplanar drawing of G. Draw the left vertex u of G at a point p_u of the outer-left set of Γ and draw the right vertex v of G at a point p_v of the outer-right set of Γ so that the segment (p_u, p_v), representing edge (u, v), doesn't intersect any of the edges of T. By Condition 3 in the definition of star-shaped drawing such two points exist. Draw the edges from u to each vertex of $L(T)$ and the edges from v to each vertex of $R(T)$. By definition of outer-left and outer-right set such edges do not alter the planarity of the drawing. □

Lemma 5.9 *Let G be an n-vertex outerplanar graph that admits an outerplanar straight-line drawing with $f(n)$ area. We have that its dual tree T admits a planar star-shaped straight-line drawing with at most $f(n)$ area.*

Proof: Suppose you have an outerplanar drawing Φ of G. Remove from Φ the poles of G, obtaining a drawing $\Phi_{I(G)}$ of the internal subgraph $I(G)$ of G. For each vertex $v_i \in I(G)$ let t be the node of T such that $\gamma^{-1}(v_i) = t$ and let p, t_l, and t_r be the parent, the left child, and the right child of t in T, respectively. Remove from Φ all the edges incident on v_i, but for $(v_i, \gamma(p))$, $(v_i, \gamma(t_l))$, and $(v_i, \gamma(t_r))$. By Lemma 5.7 the resulting drawing Φ' is a drawing of T and the area of Φ' is less or equal than the area of Φ.

Clearly, such a drawing is planar, straight-line, and order-preserving. To prove that Φ' is a a star-shaped drawing, it is sufficient to observe that Φ' satisfies Conditions 1, 2, otherwise augmenting it again by inserting straight-line edges from each node $t \in T$ to each node of the left and right polygons of the neighbors of t would result in a non-planar drawing, contradicting the hypothesis that Φ is planar. Further, Φ' satisfies Condition 3, since the placement of the poles in Φ is as required by such a condition. □

You can see in Fig. 5.9 an example of augmentation of a star-shaped drawing of a tree T to the drawing of the outerplanar graph G to which T is dual.

5.4 Outerplanar Drawings of Complete and Balanced Outerplanar Graphs

We apply the lemmata of Sect. 5.3 to construct a linear-area straight-line outerplanar drawing of a complete outerplanar graph. We denote by T_h a complete binary tree, by r_h its root, and by Γ_h its drawing. What follows is an inductive algorithm to construct a star-shaped drawing of a complete binary tree. The algorithm draws Γ_h exploiting Γ_{h-1}, as follows.

- ***Base case:*** if $h = 1$, then place r_1 in $(0, 0)$.

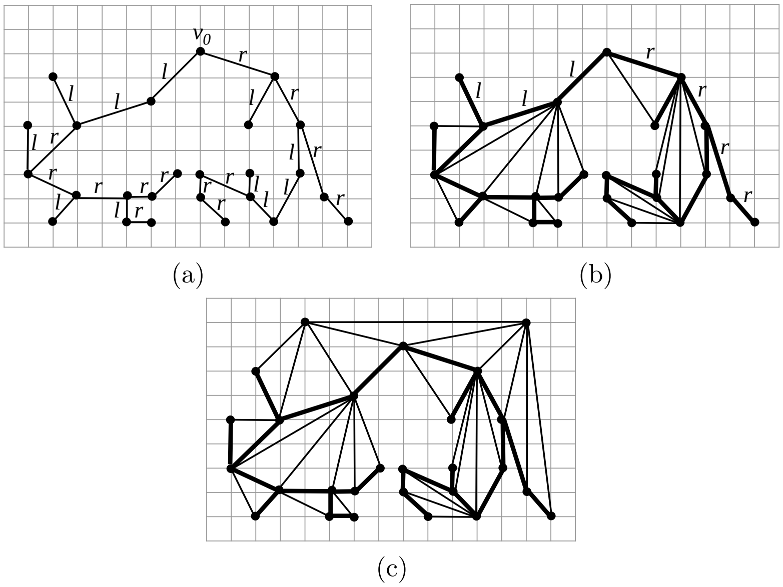
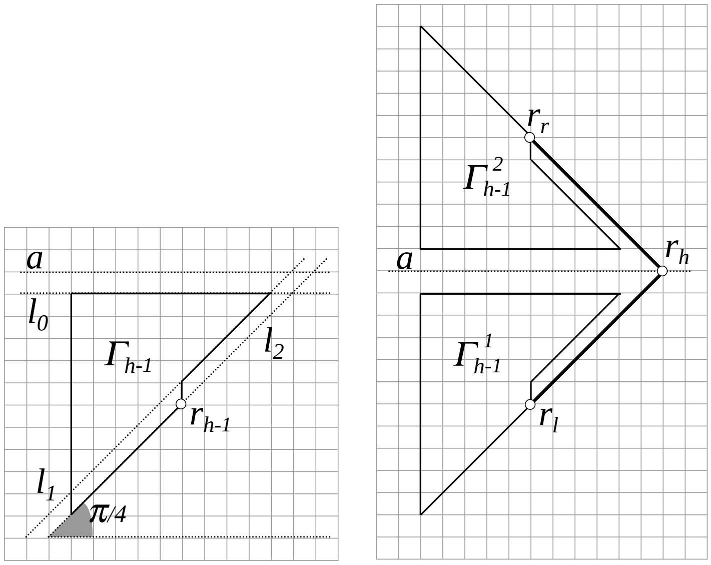


Figure 5.9: (a) A star-shaped drawing Γ of T. (b) The drawing Γ' of the internal subgraph of G obtained from Γ. (c) An outerplanar drawing of G obtained from Γ'.

- ***Inductive case:*** if $h > 1$, inductively suppose to have a star-shaped drawing Γ_{h-1} of T_{h-1} such that: (i) if $h-1$ is even, then the convex hull of Γ_{h-1} is an isosceles rectangular triangle, whose catheti C_1 and C_2 lie on lines l_1 and l_2 with slopes $3\pi/4$ and $\pi/4$, respectively; C_1 and C_2 intersect at vertex r_{h-1}, that is the rightmost point of the drawing; all the vertices of $R(T_{h-1})$ lie on C_1 and all the vertices of $L(T_{h-1})$ lie on C_2. Only vertices of $L(T_{h-1})$ and of $R(T_{h-1})$ lie on C_1 and C_2. (ii) if $h-1$ is odd, then all the vertices of $L(T_{h-1})$ lie on a line l_2 with slope $\pi/4$; vertex r_{h-1} is the vertex of such a path with the greatest y-coordinate; all the vertices of $R(T_{h-1})$, but for r_{h-1}, lie on a line l_1 with slope $\pi/4$; line l_1 is obtained by shifting of $(-1, 0)$ line l_2; the right child of vertex r_{h-1} has y-coordinate smaller than the one of any other vertex of the rightmost path of T_{h-1}, but for r_{h-1} that is one unit below it, on the same vertical line. Only vertices of $L(T_{h-1})$ are placed on l_2 and only the vertices of $R(T_{h-1})$ are placed on l_1 above r_{h-1}.

If h is even, let l_0 be the highest horizontal line intersecting Γ_{h-1}. Let a be the horizontal line one unit above l_0. Let l_2 be the lowest line with slope $\frac{\pi}{4}$ intersecting Γ_{h-1}. Reflect Γ_{h-1} with respect to a. Place r_h at the intersection between a and l_2. Insert the edges from r_h to its children (see Fig. 5.10).

If h is odd, let l_1 be the highest line with slope $\frac{3\pi}{4}$ intersecting Γ_{h-1}. Let a be the line with slope $\frac{3\pi}{4}$ two vertical units above l_1. Let l_2 be the lowest line with slope $\frac{\pi}{4}$ intersecting Γ_{h-1}. Reflect Γ_{h-1} with respect to a. Translate the new part of the drawing by a vector $(-1, 0)$. Place r_h at the intersection between a and l_2. Insert the edges from r_h to its children (see Fig. 5.11).



(a) (b)

Figure 5.10: Inductive case, h even. (a) Γ_{h-1}. (b) Γ_h.

It's easy to see that the inductive construction of the algorithm builds drawings satisfying the geometric hypotheses. A drawing produced by the algorithm is shown in Fig. 5.12 (a).

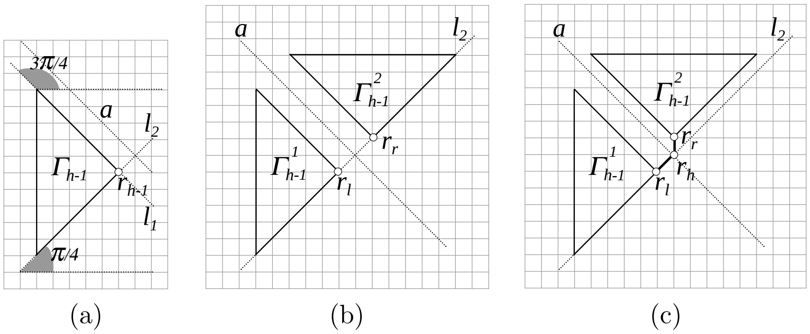


Figure 5.11: Inductive case, h odd: (a) Γ_{h-1}. (b) Γ_{h-1} after the reflection. (c) Γ_h.

We sketch the proof that the algorithm constructs a drawing Γ_h of T_h that is star-shaped, when augmented by an extra vertical line one unit to the right of the rightmost vertical line intersecting Γ_h. Denote by Γ^1_{h-1} (by Γ^2_{h-1}) the sub-drawing of Γ_h corresponding to Γ_{h-1} (resp. obtained by reflecting and, eventually, shifting Γ_{h-1}) in the inductive case of the algorithm for constructing Γ_h. Drawing Γ_h is straight-line and order-preserving by construction. The planarity of Γ_h is easily proved inductively. Namely, suppose that Γ_{h-1} is planar. Reflecting Γ^1_{h-1} with respect to line a external to Γ^1_{h-1} constructs Γ^2_{h-1}, whose planarity is a consequence of the planarity of Γ^1_{h-1}; further, Γ^1_{h-1} and Γ^2_{h-1} are separated by line a (even after translating Γ^2_{h-1} by a vector $(-1, 0)$ if h is odd) and hence they do not intersect. By the geometric inductive hypotheses of the algorithm, it's easy to observe that the planarity of the drawing is maintained by the insertion of r_h and of its incident edges. We prove that Condition 1 of a star-shaped drawing is satisfied by Γ_h. Inductively suppose that Γ_{h-1} satisfies Condition 1 of a star-shaped drawing. Then, after reflecting Γ^1_{h-1} with respect to line a and, if h is odd, after translating Γ^2_{h-1} by a vector $(-1, 0)$, Condition 1 of a star-shaped drawing is satisfied by all the vertices in Γ^1_{h-1} and in Γ^2_{h-1}, since reflection and translation preserve the shapes of the drawings and since Γ^1_{h-1} and Γ^2_{h-1} do not overlap being separated by line a. It remains to show that, if $m \geq 2$ (if $p \geq 2$), the left cycle of the neighbors $(r_h, v_1, \dots, v_m)$ (the right cycle of the neighbors $(r_h, u_1, \dots, u_p)$) of r_h is represented in Γ_h by a left polygon of the neighbors $P_l(r_h)$ (by a right polygon of the neighbors $P_r(r_h)$) such that, if $m > 2$ (if $p > 2$), each segment (r_h, v_i), with $2 \leq i \leq m-1$ (each segment (r_h, u_j), with $2 \leq j \leq p-1$)

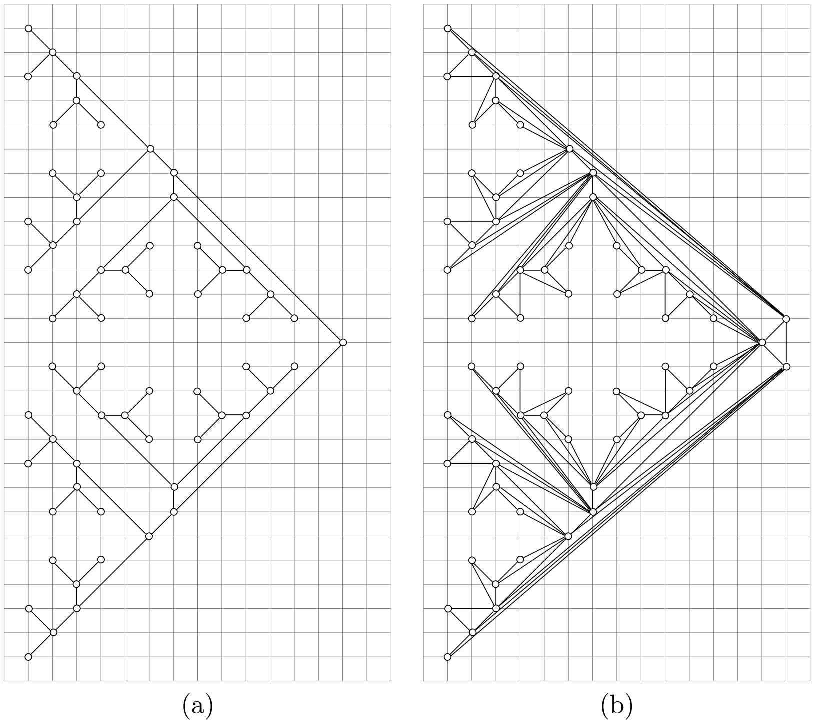


Figure 5.12: An application of the algorithm in *Sect. 5.4*. (a) Γ_6. (b) The drawing of the outerplanar graph G_6 constructed from Γ_6.

belongs to the interior of $P_l(r_h)$ ($P_r(r_h)$), but for its endpoints r_h and v_i (r_h and u_j). Suppose h is even. Let l_2 be the line with slope $\pi/4$ through r_h. By the geometric inductive hypotheses of the algorithm, the left child r_l of r_h lies on l_2, the rightmost path of r_l lies on a line l_1 obtained by shifting of $(-1, 0)$ line l_2, the right child of vertex r_l is the vertex of the rightmost path of T_{h-1} with smallest y-coordinate, and it is one vertical unit above r_l. Hence, if $m \geq 2$, $P_l(r_h)$ is a trapezium with parallel sides lying on l_2 and l_1; hence, $P_l(r_h)$, and symmetrically $P_r(r_h)$, is a convex polygon satisfying Condition 1 of a star-shaped drawing. If h is odd, let a be the line with slope $3\pi/4$ through r_h. By the geometric inductive hypotheses of the algorithm, the rightmost path of the left child r_l of r_h lies on a line l_1 with slope $3\pi/4$ obtained by shifting of $(-1,-1)$ line a; hence $P_l(r_h)$, and analogously $P_r(r_h)$, is a triangle, that is a convex shape satisfying Condition 1 of a star-shaped drawing. Now we prove that Γ_h satisfies Condition 2 of a star-shaped drawing. Again, this can be done inductively. Namely, suppose that Γ_{h-1} satisfies Condition 2 of a star-shaped drawing. No left or right polygon of the neighbors of a vertex in Γ^1_{h-1} overlaps any left or right polygon of the neighbors of a vertex in Γ^2_{h-1}, since such polygons are separated by line a. It remains to show that the left and right polygons of the neighbors of r_h, say $P_l(r_h)$ and $P_r(r_h)$, respectively, do not overlap with any left or right polygon of the neighbors. First, $P_l(r_h)$ does not overlap with $P_r(r_h)$ or with any left or right polygon of the neighbors of a vertex in Γ^2_{h-1}, since such polygons are separated by line a. Analogously, $P_r(r_h)$ does not overlap with any left or right polygon of the neighbors of a vertex in Γ^1_{h-1}, since such polygons are separated by line a. If h is odd, then $P_l(r_h)$ is contained in the closed half-plane above line l_1; since only the vertices of the rightmost path of the subtree of T_h rooted at r_l lie on l_1 or above it, then $P_l(r_h)$ does not overlap with any left or right polygon of the neighbors of a vertex in Γ^1_{h-1}. It can be proved analogously that $P_r(r_h)$ does not overlap with any left or right polygon of the neighbors of a vertex in Γ^2_{h-1}. If h is even the closed region of the plane delimited by l_1, l_2, a, and the line through r_l and its right child contains only the vertices of the rightmost path of r_l. Since such a region contains $P_l(r_h)$, then $P_l(r_h)$ does not overlap with any left or right polygon of the neighbors of a vertex in Γ^1_{h-1}. It can be proved analogously that $P_r(r_h)$ does not overlap with any left or right polygon of the neighbors in Γ^2_{h-1}.

Now we prove that Γ_h augmented by a vertical line one unit to the right of r_h satisfies Condition 3 of a star-shaped drawing. Draw p_u at the point one unit to the right and one unit below r_h and draw p_v at the point one unit to the right and one unit above r_h. Since the leftmost and rightmost paths of T_h lie entirely on lines with slopes $\pi/4$ or $3\pi/4$ and p_u and p_v are in the opposite part of the drawing with respect to such lines, then straight-line segments can connect p_u to each vertex of $L(T_h)$ and p_v to each vertex of $R(T_h)$ without introducing crossings.

Now we analyze the area requirements of the drawings constructed by the above algorithm. Let $height_h$ and $width_h$ be the height and the width of Γ_h, respectively. We distinguish two cases:

1. ***h is even:*** Since Γ_h is obtained by reflecting Γ^1_{h-1} with respect to the horizontal line a, then $height_h$ is twice $height_{h-1}$ plus one unit corresponding to a; hence, (1) $height_h = 2 \cdot height_{h-1} + 1$. The width of Γ_h

is given by the width of Γ^1_{h-1} plus the horizontal displacement of r_h with respect to rightmost line l intersecting Γ^1_{h-1}. It's easy to see that r_h is two units to the right of l, hence (2) $width_h = width_{h-1} + 2$.

2. ***h is odd:*** The height of Γ_h is equal to the height of Γ^1_{h-1} plus the vertical displacement of the highest line intersecting Γ^2_{h-1} with respect to the highest line intersecting Γ^1_{h-1}. It's easy to see that such a displacement is two grid units; hence, (3) $height_h = height_{h-1} + 2$. The width of Γ_h is given by the width of Γ^2_{h-1} (that is the height of Γ_{h-1}, due to the reflection) plus the horizontal displacement of the leftmost line intersecting Γ^1_{h-1} with respect to leftmost line intersecting Γ^2_{h-1}. It's easy to see that such a displacement is one grid unit; hence, (4) $width_h = height_{h-1} + 1$.

Let us compute $height_h$ as a function of h and, consequently, as a function of n. Suppose h is even. Substituting (3) into (1) we get $height_h = 2 \cdot height_{h-2} + 5$. Using $height_2 = 3$, a simple inductive proof shows that (5) $height_h = 4 \cdot 2^{\frac{h}{2}} - 5$, which is $4\sqrt{n-1} - 5$, since $h = \log_2(n-1)$. If h is odd, we derive the value of $height_h$ by substituting (5) into (3). Namely, (6) $height_h = 4 \cdot 2^{\frac{h-1}{2}} - 5 + 2 = \frac{4}{\sqrt{2}} \cdot 2^{\frac{h}{2}} - 3$. Hence, $height_h = \frac{4}{\sqrt{2}}\sqrt{n-1} - 3$.

Now we compute $width_h$. Suppose h is odd. Substituting (5) into (4), we get (7) $width_h = 4 \cdot 2^{\frac{h-1}{2}} - 5 + 1 = \frac{4}{\sqrt{2}} \cdot 2^{\frac{h}{2}} - 4$ and hence $width_h = \frac{4}{\sqrt{2}}\sqrt{n-1} - 4$. Finally, suppose h is even. Substituting (7) into (2), we get (8) $width_h = \frac{4}{\sqrt{2}} \cdot 2^{\frac{h-1}{2}} - 4 + 2 = 2 \cdot 2^{\frac{h}{2}} - 2$ and hence $width_h = 2\sqrt{n-1} - 2$.

We exploit the above algorithm and Lemma 5.8 to prove the following theorem.

Theorem 5.3 *Given an n-vertex complete outerplanar graph G_h with height h, there exists an $O(n)$-time algorithm that constructs an outerplanar straight-line grid drawing Γ_h of G_h such that:*

1. *if h is even, then the height of Γ_h is $4\sqrt{n-1}-5$ and its width is $2\sqrt{n-1}-1$;*

2. *if h is odd, then the height of Γ_h is $\frac{4}{\sqrt{2}}\sqrt{n-1} - 3$ and its width is $\frac{4}{\sqrt{2}}\sqrt{n-1} - 4$;*

3. *isomorphic subgraphs of G_h have congruent drawings in Γ_h up to a translation and a reflection; and*

4. Γ_h *is axially symmetric.*

Proof: Denote by T_h the dual tree of G_h. Drawing Γ_h can be constructed as follows. First, we apply the algorithm shown in this section to obtain a star-shaped drawing Γ_h of T_h. Second, using Lemmata 5.6 and 5.8, we construct a straight-line outerplanar drawing $\Gamma_{I(G)}$ of the internal subgraph of G_h. Finally, we draw the poles of G_h and their incident edges, obtaining Γ_h. This is done as follows.

Draw the left vertex one unit below and one unit to the right of r_h, and draw the right vertex one unit above and one unit to the right of r_h. As already discussed, this placement allows to draw straight-line edges from the left vertex to each node of $L(T_h)$ and from the right vertex to each node of $R(T_h)$ without introducing crossings. Moreover, this placement increases by at most one unit the width of the drawing (only if h is even) and doesn't alter the height of Γ_h. See Fig. 5.12 (b).

The bounds on the height and on the width of Γ_h easily descend from the bounds given for the star-shaped drawings of complete binary trees. Unfortunately, the drawings obtained by the described algorithm have poor angular resolution. Namely, some calculations show that there are angles in the drawing that decrease as fast as $\frac{k}{n}$, with k constant. □

From the fact that a balanced outerplanar graph can be augmented to complete by adding a linear number of nodes and without altering its height we have:

Theorem 5.4 *Given an n-vertex balanced outerplanar graph G, there exists an $O(n)$-time algorithm that constructs an outerplanar straight-line grid drawing Γ of G such that both the height and the width of Γ are $O(\sqrt{n})$.*

5.5 Outerplanar Drawings of Outerplanar Graphs in $O(n^{1.48})$ Area

This section is devoted to the proof of the following theorem.

Theorem 5.5 *Given an n-vertex outerplanar graph G, there exists an $O(n)$-time algorithm that constructs an $O(n^{1.48})$ area outerplanar straight-line grid drawing of G.*

The main ingredients of the proof are: (i) a recursive algorithm for constructing a star-shaped drawing of a binary tree, (ii) Lemma 5.8, and (iii) Lemma 5.10 presented by Chan in [Cha02].

Lemma 5.10 *(Chan [Cha02]) Let* $p = 0.48$*. Given any binary tree* T *of size* n*, there exists a root-to-leaf path* π *such that for any left subtree* α *and right subtree* β *of* π*,* $|\alpha|^p + |\beta|^p \leq (1-\delta)n^p$*, for some constant* $\delta > 0$*.*

First, we show two techniques, called Constructions 1–2, for constructing a star-shaped drawing Γ_i, with $i \in \{1, 2\}$, of a general binary tree T with n nodes. Each one is defined in terms of itself and of the other one. Let $S = (v_0, v_1, \ldots, v_m)$ be any spine of T.

Let s_i be the non-spine child of v_i and let $T(s_i)$ be the subtree of T rooted at s_i. We denote by $W_1(T)$ (by $W_2(T)$) the width of Γ_1 (of Γ_2). We denote by $W_1(n)$ (by $W_2(n)$) the maximum width of a drawing obtained from an application of Construction 1 (of Construction 2) among all possible n-node input binary trees. Clearly, $W_i(T) \leq W_i(n)$.

Now we show **C**onstruction 1. First, we draw each $v_i \in S$ together with $T(s_i)$, obtaining $\Gamma(v_i)$; then we put all the $\Gamma(v_i)$ together to obtain Γ_1. Construction 1 has four subcases, labeled $1xy$, $x \in \{t, b\}$ and $y \in \{l, r\}$. Index x states that S is drawn going towards the top ($x = t$) or towards the bottom ($x = b$) of Γ_1. Index y states that $L(T)$ ($y = l$), or $R(T)$ ($y = r$), is drawn going towards the left of Γ_1. In the following we show the details of Construction $1bl$, while the others are easily obtained from $1bl$ after a reflection with respect to the x-axis and/or a switch of the left with the right and vice-versa. Construction 1 is shown in Fig. 5.13.

Suppose v_1 is the right child of v_0. Let k be the first index such that v_k is the left child of v_{k-1}. In the following we denote the subtree $T(s_0)$ also with $T(s_l)$ and we denote the subtree $T(s_{k-1})$ also with $T(s_r)$. Draw $T(s_0)$ and $T(s_{k-1})$ with Construction $1bl$, obtaining $\Gamma(s_0)$ and $\Gamma(s_{k-1})$, respectively. Draw v_0 one unit above and one unit to the right of $B(\Gamma(s_0))$, obtaining $\Gamma(v_0)$. Draw v_{k-1} one unit above and one unit to the left of $B(\Gamma(s_{k-1}))$, obtaining $\Gamma(v_{k-1})$. Draw any other left subtree of S with Construction $2tr$ and any other right subtree of S with Construction $2bl$, obtaining $\Gamma(s_i)$. If s_i is the left child of v_i, draw v_i on the same row and one unit to the right of s_i, else (s_i is the right child of v_i) draw v_i on the same horizontal row and one unit to the left of s_i, obtaining $\Gamma(v_i)$.

Now suppose v_1 is the left child of v_0. Let k be the first index such that v_k is the right child of v_{k-1}. In the following we denote the subtree $T(s_0)$ also with $T(s_r)$ and we denote the subtree $T(s_{k-1})$ also with $T(s_l)$. Draw $T(s_0)$ and $T(s_{k-1})$ with Construction $1bl$, obtaining $\Gamma(s_0)$ and $\Gamma(s_{k-1})$, respectively. Draw v_0 one unit above and one unit to the left of $B(\Gamma(s_0))$, obtaining $\Gamma(v_0)$.

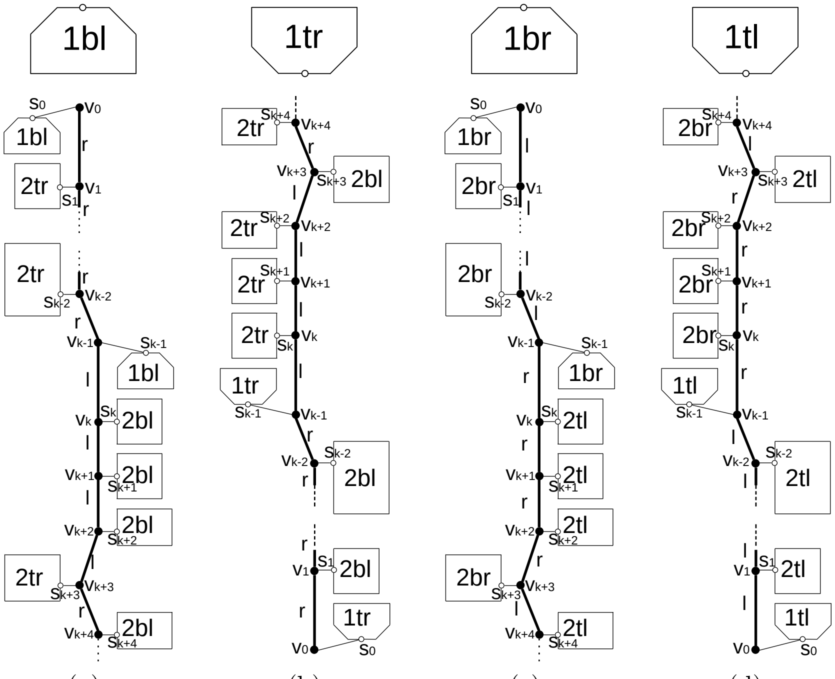


Figure 5.13: Constructions (a) $1bl$, (b) $1tr$, (c) $1br$ and (d) $1tl$. The edges (v_i, v_{i+1}) labeled r (l) are such that v_{i+1} is the right (resp. left) child of v_i. The thick edges show the spine of T.

Draw v_{k-1} one unit above and one unit to the right of $B(\Gamma(s_{k-1}))$, obtaining $\Gamma(v_{k-1})$. Draw any other left subtree with Construction $2tr$ and any other right subtree with Construction $2bl$, obtaining $\Gamma(s_i)$. If s_i is the left child of v_i, draw v_i on the same row and one unit to the right of s_i, else (s_i is the right child of v_i) draw v_i on the same row and one unit to the left of s_i, obtaining $\Gamma(v_i)$.

Now we put together all the $\Gamma(v_i)$, $0 \leq i \leq m$ as follows. Place $\Gamma(v_0)$ anywhere in the plane. For $1 \leq i \leq m$ do as follows:

- if v_i is the left child of v_{i-1} and v_{i+1} is the left child of v_i, then draw $\Gamma(v_i)$ so that v_i is on the same column of v_{i-1} and so that $b(\Gamma(v_{i-1}))$ is one unit above $t(\Gamma(v_i))$.

- if v_i is the right child of v_{i-1} and v_{i+1} is the right child of v_i, then draw $\Gamma(v_i)$ so that v_i is on the same column of v_{i-1} and so that $b(\Gamma(v_{i-1}))$ is one unit above the $t(\Gamma(v_i))$.

- if v_i is a leaf ($i = m$), then draw $\Gamma(v_i)$ so that v_i is on the same column of v_{i-1} and so that $b(\Gamma(v_{i-1}))$ is one unit above v_i.

- if v_i is the left child of v_{i-1} and v_{i+1} is the right child of v_i then draw v_i on the column one unit to the left with respect to the column of v_{i-1} and so that $b(\Gamma(v_{i-1}))$ is one unit above $t(\Gamma(v_i))$.

- if v_i is the right child of v_{i-1} and v_{i+1} is the left child of v_i then draw v_i on the column one unit to the right with respect to the column of v_{i-1} and so that $b(\Gamma(v_{i-1}))$ is one unit above $t(\Gamma(v_i))$.

Property 5.3 *Construction 1bl guarantees that all the vertices of $L(T)$ are visible from any point that is above and to the left of $B(\Gamma_1)$ and that all the vertices of $R(T)$ are visible from any point that is above and to the right of $B(\Gamma_1)$.*

Property 5.4 *Suppose that Construction 2tr constructs a star-shaped drawing in which the leftmost and the rightmost paths of the tree lie on the right side of the drawing's bounding box. Suppose also that Construction 2bl constructs star-shaped drawings in which the leftmost and the rightmost paths of the tree lie on the left side of the drawing's bounding box. We have that the drawing obtained with Construction 1bl is star-shaped.*

Property 5.5 *The following equality holds:*

$$\begin{aligned} W_1(T) = 2 \quad &+ \quad \max\{W_1(T(s_l)), \max_i\{W_2(T(s_i))\}\} + \\ &+ \quad \max\{W_1(T(s_r)), \max_j\{W_2(T(s_j))\}\}, \end{aligned}$$

where i is such that s_i is the left child of v_i and j is such that s_j is the right child of v_j, with $i, j \notin \{0, k-1\}$.

Analogous properties hold for Constructions $1br$, $1tl$,and $1tr$.

Construction 2 is as follows. We have four subcases, say $2xy$, where $x \in \{t, b\}$ and $y \in \{l, r\}$. Index x states that $L(T)$ is drawn going towards the top ($x = t$) or going towards the bottom ($x = b$) of Γ_2. Index y states that the root is drawn on the right side ($y = r$) or on the left side ($y = l$) of Γ_2. In the following we show Construction $2bl$, while the other cases are easily obtained from $2bl$ after a reflection with respect to the y-axis and/or a switch of the left with the right and vice-versa. Construction 2 is shown in Fig. 5.14.

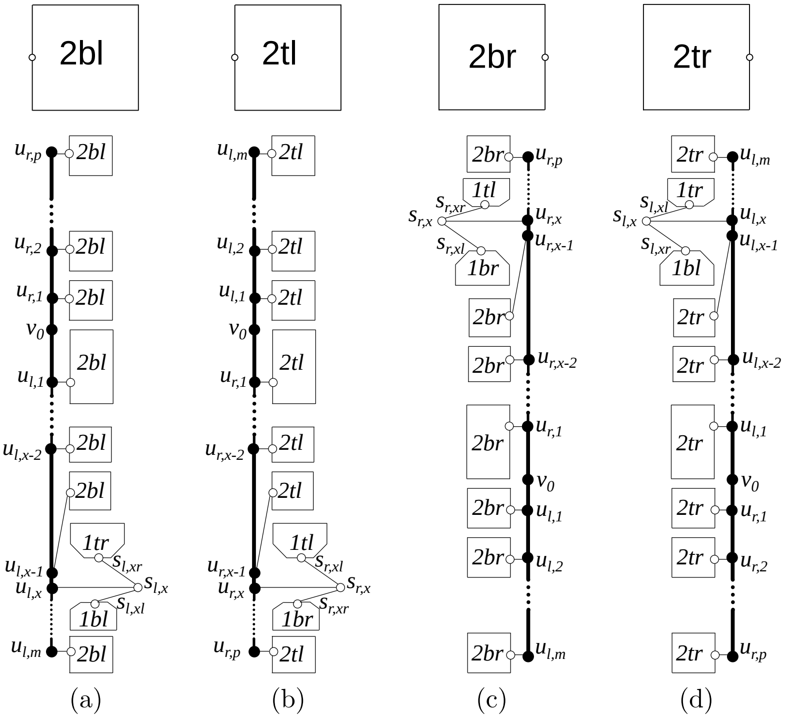


Figure 5.14: Constructions (a) $2bl$, (b) $2tl$, (c) $2br$, (d) $2tr$. The thick edges show $L(T)$ and $R(T)$.

Let v_0 be the root of T, let $C_l = (u_{l,0}, u_{l,1}, \ldots, u_{l,m})$ be $L(T)$, and let $C_r = (u_{r,0}, u_{r,1}, \ldots, u_{r,p})$ be the rightmost path of T, with $u_{l,0} = u_{r,0} = v_0$. Let $s_{l,i}$ be the right child of a node $u_{l,i} \in C_l$ and let $s_{r,i}$ be the left child of a node $u_{r,i} \in C_r$; we call $T(s_{l,i})$ the subtree of T rooted at $s_{l,i}$ and $T(s_{r,i})$

the subtree of T rooted at $s_{r,i}$. First, we draw each $u_{l,i} \in C_l$ together with $T(s_{l,i})$ and each $u_{r,i} \in C_r$ together with $T(s_{r,i})$, obtaining $\Gamma(u_{l,i})$ and $\Gamma(u_{r,i})$ respectively; then we put all the $\Gamma(u_{l,i})$ and the $\Gamma(u_{r,i})$ together to obtain Γ_2.

Let $s_{k,x}$ be the one among nodes $s_{j,i}$, with $j = l$ and $i = 1, 2, \dots, m$ and with $j = r$ and $i = 1, 2, \dots, p$, such that $T(s_{k,x})$ has the maximum number of nodes among all the subtrees $T(s_{j,i})$. Let $T(s_{k,xl})$ and $T(s_{k,xr})$ be the left and the right subtree of $s_{k,x}$, with roots $s_{k,xl}$ and $s_{k,xr}$, respectively. Draw $T(s_{k,xl})$ with Construction $1bl$ and draw $T(s_{k,xr})$ with Construction $1tr$, obtaining $\Gamma(s_{k,xl})$ and $\Gamma(s_{k,xr})$, respectively. Draw any other subtree $T(s_{j,i})$ with Construction $2bl$, obtaining $\Gamma(s_{j,i})$.

Place $\Gamma(s_{k,xl})$ anywhere in the plane. Place $\Gamma(s_{k,xr})$ so that $b(\Gamma(s_{k,xr}))$ is three vertical units above $t(\Gamma(s_{k,xl}))$ and so that $l(\Gamma(s_{k,xr}))$ is on the same column of $l(\Gamma(s_{k,xl}))$. If $k = l$, place $s_{l,x}$ one unit above $t(\Gamma(s_{l,xl}))$ and one unit to the right of the rightmost boundary between $r(\Gamma(s_{l,xl}))$ and $r(\Gamma(s_{l,xr}))$. Draw $u_{l,x}$ on the same row of $s_{l,x}$, one unit to the left of $l(\Gamma(s_{l,xl}))$. Draw $u_{l,x-1}$ one unit above $u_{l,x}$. If $k = r$, place $s_{r,x}$ one unit below $b(\Gamma(s_{r,xr}))$ and one unit to the right of the rightmost boundary between $r(\Gamma(s_{r,xl}))$ and $r(\Gamma(s_{r,xr}))$. Draw $u_{r,x}$ on the same row of $s_{r,x}$, one unit to the left of $l(\Gamma(s_{r,xr}))$. Draw $u_{r,x-1}$ one unit below $u_{r,x}$. Place $\Gamma(s_{k,x-1})$ so that $l(\Gamma(s_{k,x-1}))$ is on the same column of $l(\Gamma(s_{k,x}))$ and so that (if $k = l$) $b(\Gamma(s_{k,x-1}))$ is one unit above $t(\Gamma(s_{k,x}))$ or (if $k = r$) $t(\Gamma(s_{k,x-1}))$ is one unit below $b(\Gamma(s_{k,x}))$, obtaining $\Gamma(u_{k,x-1})$.

For each $\Gamma(s_{j,i})$, with $j = l$ and $i = 1, 2, \dots, m$ and with $j = r$ and $i = 1, 2, \dots, p$, but for $\Gamma(s_{k,x-1})$ and $\Gamma(s_{k,x})$, place $u_{j,i}$ one unit to the left of $s_{j,i}$, obtaining $\Gamma(u_{j,i})$. Place all the $\Gamma(u_{l,i})$ (eventually also $\Gamma(u_{k,x-1})$ if $k = l$) so that all $u_{l,i}$ are on the same column, so that $t(\Gamma(u_{l,i}))$ is one unit below $b(\Gamma(u_{l,i-1}))$, with $2 \leq i \leq m$, obtaining Γ_l. Place all the $\Gamma(u_{r,i})$ (eventually also $\Gamma(u_{k,x-1})$ if $k = l$) so that all $u_{r,i}$ are on the same column, so that $b(\Gamma(u_{r,i}))$ is one unit above $t(\Gamma(u_{r,i-1}))$, with $2 \leq i \leq p$, obtaining Γ_r.

Notice that if $x = 1$ then v_0 is already drawn in Γ_l (if $k = l$) or in Γ_r (if $k = r$). Otherwise, augment Γ_l by drawing v_0 on the same column of vertices $u_{l,i}$, one unit above $t(\Gamma(u_{l,1}))$.

Finally, place Γ_l and Γ_r together, so that $l(\Gamma_l)$ and $l(\Gamma_r)$ are on the same column, and so that $b(\Gamma_r)$ is one unit above $t(\Gamma_l)$.

Property 5.6 *Construction $2bl$ guarantees that all the vertices of $L(T)$ are on the left side of the bounding box of Γ_2. Construction $2bl$ guarantees that all the vertices of the rightmost path of T are on the left side of the bounding box of Γ_2.*

Property 5.7 *Suppose that Constructions* $1tr$ *and* $1bl$ *construct star-shaped drawings. Suppose that Construction* $1tr$ *is such that* $L(T(s_{k,xr}))$ *is visible from any point that is below and to the right of* $B(\Gamma(s_{k,xr}))$. *Suppose that Construction* $1tr$ *is such that* $R(T(s_{k,xr}))$ *is visible from any point that is below and to the to the left of* $B(\Gamma(s_{k,xr}))$. *Suppose that Construction* $1bl$ *is such that* $L(T(s_{k,xl}))$ *is visible from any point that is above and to the left of* $B(\Gamma(s_{k,xl}))$. *Suppose that Construction* $1bl$ *is such that* $R(T(s_{k,xl}))$ *is visible from any point that is above and to the right of* $B(\Gamma(s_{k,xl}))$. *We have that a drawing constructed by Construction* $2bl$ *is star-shaped.*

Property 5.8 *The following equality holds:*

$$W_2(T) = \max\{2 + W_1(T(s_{k,xl})), 2 + W_1(T(s_{k,xr})), \max\{1 + W_2(T(s_{j,i}))\}\},$$

where $s_{j,i} \neq s_{k,x}$.

Analogous properties hold for Constructions $2br$, $2tl$,and $2tr$.

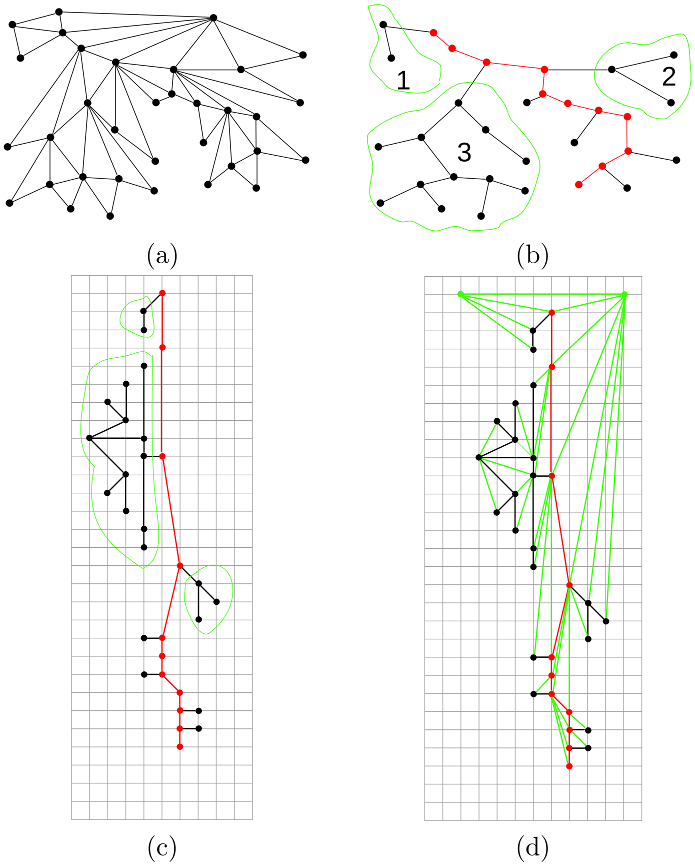
We can use Constructions 1–2 for constructing a star-shaped drawing Γ of a binary tree T as follows. First, we select any spine. Second, we apply Construction $1bl$. Third, we recursively apply all the constructions in the appropriate cases. From the above properties we have that Γ is star-shaped.

At this point we can draw any outerplanar graph G with dual tree T as follows. First, we draw T with the above algorithm. Second, we apply Lemma 5.8 to construct an outerplanar drawing of the internal subgraph of G with the same height and width of T. Third, exploiting Property 5.3 we place the poles of G obtaining a drawing that has the same height and width plus one unit and two units, respectively.

An example of application of the algorithm is shown in Fig. 5.15.

Now we analyze the height and the width of Γ. About the height, it's easy to see that there is at least one vertex for each horizontal line that intersects Γ. So we immediately obtain that the height of Γ is $O(n)$.

Now we focus on the width $W(T)$ of Γ. To get a good bound for $W(T)$, we need to carefully choose the spine of T (and recursively the spine of each subtree drawn with Construction 1) while applying the algorithm. Denote by $W(n)$ the maximum width of a drawing obtained from an application of the algorithm among all possible n-node input binary trees. Observe that since the algorithm starts by applying Construction 1, then $W(n) = W_1(n)$. By definition we have $W(T) \leq W(n)$.



(a) (b) (c) (d)

Figure 5.15: Application of the algorithm in Sect. 5.5. (a) An outerplanar graph G. (b) The dual tree T of G and its subdivision by the selection of its spine. (c) The star-shaped drawing of T. (d) The outerplanar drawing of G .

Let n_1 (n_2) be the number of vertices of the heaviest left (right) subtree of the spine S of T, that is, the number of vertices of the left (right) subtree of S with the maximum number of nodes. Spine S can be chosen so that, for any left subtree α and right subtree β of S, we have $|\alpha|^{0.48} + |\beta|^{0.48} \leq (1-\delta)n^{0.48}$, for some constant $\delta > 0$, according to Lemma 5.10 appeared in [Cha02]. Clearly, the previous inequality implies that $n_1^{0.48} + n_2^{0.48} \leq (1-\delta)n^{0.48}$. Observe that, by definition, $W(n)$, $W_1(n)$, and $W_2(n)$ are non-decreasing functions of n.

We claim that $W_1(n) \leq 7 \cdot n^{0.48} - 6$ and that $W_2(n) \leq 7 \cdot n^{0.48} - 4$. In the following, we prove such claims by induction on n. In the base case $n = 1$ and the trivial drawing of a single node satisfies both inequalities. Now suppose that the claims hold for trees with at most $n - 1$ nodes.

First, we prove that $W_1(n) \leq 7 \cdot n^{0.48} - 6$. By property 5.5, we have:

$$\begin{aligned} W_1(T) = 2 \quad &+ \quad \max\{W_1(T(s_l)), \max_i\{W_2(T(s_i))\}\} + \\ &+ \quad \max\{W_1(T(s_r)), \max_j\{W_2(T(s_j))\}\}, \end{aligned}$$

where i is such that s_i is the left child of v_i and j is such that s_j is the right child of v_j, with $i, j \notin \{0, k-1\}$. Observe that $T(s_l)$ and each of $T(s_i)$, where i is such that s_i is the left child of v_i, have at most n_1 nodes, and that $T(s_r)$ and each of $T(s_j)$, where j is such that s_j is the right child of v_j, have at most n_2 nodes. Hence, we have $W_1(n) = 2 + \max\{W_1(n_1), W_2(n_1)\} + \max\{W_1(n_2), W_2(n_2)\}$. Applying the inductive hypothesis, we have $W_1(n) = 2 + \max\{7 \cdot n_1^{0.48} - 6, 7 \cdot n_1^{0.48} - 4\} + \max\{7 \cdot n_2^{0.48} - 6, 7 \cdot n_2^{0.48} - 4\} = 2 + 7 \cdot n_1^{0.48} - 4 + 7 \cdot n_2^{0.48} - 4 = 7 \cdot (n_1^{0.48} + n_2^{0.48}) - 6$. Since $n_1^{0.48} + n_2^{0.48} \leq (1 - \delta)n^{0.48} \leq n^{0.48}$, the first claim follows.

We prove that $W_2(n) \leq 7 \cdot n^{0.48} - 4$. By property 5.8, we have $W_2(T) = \max(2 + W_1(T(s_{k,xl})), 2 + W_1(T(s_{k,xr})), 1 + \max(W_2(T(s_{j,i}))))$, where $s_{j,i} \neq s_{k,x}$. Observe that $T(s_{k,xl})$ and $T(s_{k,xr})$ have no more than $n - 1$ nodes and that $T(s_{j,i})$, where $s_{j,i} \neq s_{k,x}$, has no more than $n/2$ nodes, since $T(s_{k,x})$ is chosen to be the subtree of the leftmost and rightmost paths of T with the maximum number of nodes. Hence, we get $W_2(n) \leq \{2 + W_1(n), 1 + W_2(n/2)\}$. Applying the inductive hypothesis, we have $W_2(n) \leq \max\{2 + 7 \cdot n^{0.48} - 6, 1 + 7 \cdot (n/2)^{0.48} - 4\} = \max\{7 \cdot n^{0.48} - 4, 7 \cdot (n/2)^{0.48} - 3\}$. Hence, we have only to show that $7 \cdot (n/2)^{0.48} - 3 \leq 7 \cdot n^{0.48} - 4$. The previous inequality is satisfied if $7(1 - \frac{1}{2^{0.48}})n^{0.48} \geq 1$. However, since $n \geq 1$, we have $7(1 - \frac{1}{2^{0.48}})n^{0.48} > 1.98$.

From the results on the height and on the width, we obtain the $O(n^{1.48})$ area bound on Γ. It is easy to see that the algorithm can be implemented to run in linear time.

5.6 Outerplanar Drawings of Outerplanar Graphs in $O(dn \log n)$ Area

In this section we show an algorithm for constructing straight-line outerplanar drawings of outerplanar graphs with degree d in $O(dn \log n)$ area.

Similarly to the algorithm shown in the previous section, the algorithm we present in this section has the following steps: (i) augment the input outerplanar graph G to a maximal outerplanar graph G'; (ii) select any edge (u_l, u_r) incident to the outer face of the outerplanar embedding $\mathcal{E}$ of G' and root the dual binary tree T of $\mathcal{E}$ at the internal face r of $\mathcal{E}$ containing (u_l, u_r); construct a star-shaped drawing Γ of T; (iii) insert the poles of G' and the edges that are needed to augment Γ in a drawing Γ' of G'; (iv) remove the dummy edges inserted during the first step to obtain a drawing of G.

Differently from the algorithm shown in the previous section, some care has to be taken when augmenting the outerplanar graph to maximal. In fact, this must be done so that the degree of the graph does not asymptotically increase. However, Kant and Bodlaender proved in [KB97] that any outerplanar graph can be augmented to maximal by inserting dummy edges that do not asymptotically increase the degree of the graph.

Now we describe how to construct a star-shaped drawing Γ of T. The outline of such a construction is as follows. A path S is removed from T, together with the edges incident to the vertices of S. The subtrees that are disconnected from the removal of S are recursively drawn. Path S is chosen so that each one of such subtrees is "small", that is, has at most $n/2$ nodes. The drawings of the recursively drawn subtrees are horizontally aligned, namely they are all contained in an horizontal strip H. The vertical extension of such a strip is given by the height of the highest drawing of a subtree recursively drawn. The nodes of S are drawn in the *upper part* and in the *lower part* of the drawing, that is, in the $4d + 1$ horizontal grid lines above and below H, respectively. In particular, S is partitioned into subpaths, and each subpath "cuts" H, that is, the subpath is drawn in part above and in part below H. The drawing is constructed to satisfy the visibility properties of a star-shaped drawing. In particular, the leftmost and the rightmost path of T are placed on the lowest line intersecting the drawing.

Denote by h_1 and h_2 the horizontal grid lines delimiting H with h_1 above h_2 at a vertical distance that will be determined later. The $4d+1$ horizontal grid lines above h_1 (resp. below h_2), that compose the upper part (resp. the lower part) of the drawing, are labeled by $u_1, u_2, \dots, u_{4d+1}$ (resp. by $l_1, l_2, \dots, l_{4d+1}$)

from the lowest to the highest (resp. from the highest to the lowest). All the nodes of $L(T)$ and of $R(T)$ lie on l_{4d+1}. See Fig. 5.16.

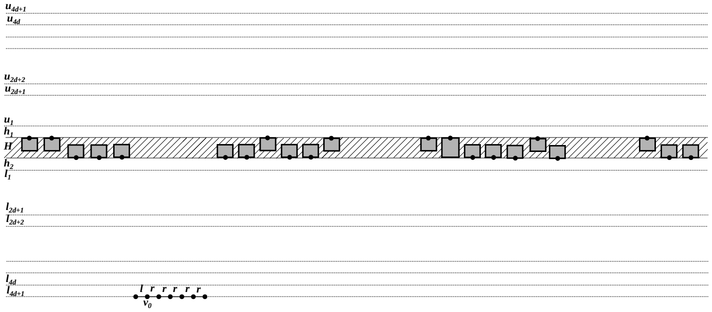


Figure 5.16: The strip H, the upper and the lower part of the drawing, the placement of $L(T)$ and $R(T)$ in Γ.

Assume T is rooted at any node v_0 of degree at most 2. Select a spine $S = (v_0, v_1, \ldots, v_m)$ in T, such that, for $1 \leq i \leq k$, v_i is the child of v_{i-1} that is root of the heaviest subtree of v_{i-1}. The nodes of T belonging to S (not belonging to S) are *spine nodes* (resp. *non-spine nodes*). We call *left edge* (*right edge*) an edge (v_{i-1}, v_i) of S such that v_i is the left child of v_{i-1} in S (resp. the right child of v_{i-1} in S). We prove that each non-spine node u_i child of a spine node v_i is root of a subtree with no more than $n/2$ nodes. Let v_{i+1} be the spine node child of v_i (notice that v_{i+1} exists, otherwise u_i would be chosen as a spine node). Suppose, for a contradiction, that $T(u_i)$ has at least $1 + n/2$ nodes. However, by the way in which the spine is chosen, $T(v_{i+1})$ has also at least $1 + n/2$ nodes. Hence, T would have more than n nodes, providing a contradiction.

Path S is partitioned into vertex-disjoint subpaths $S_0, S_1, \ldots, S_q$, so that, for $0 \leq j \leq q$, path $S_j = (v^j_k, v^j_{k+1}, \ldots, v^j_l, v^j_{l+1}, \ldots, v^j_{f-1}, v^j_f)$ is defined as follows::

- $v^0_k = v_0$, and, for $1 \leq j \leq q$, v^j_k is the node after v^{j-1}_f in S.
- If (v^j_k, v^j_{k+1}) is a right edge, then let v^j_l be the first node after v^j_k in S such that (v^j_l, v^j_{l+1}) is a left edge.
  - If (v^j_{l+1}, v^j_{l+2}) is a left edge then let v^j_f be the first node after v^j_l in S such that (v^j_f, v^j_{f+1}) is a right edge.

  - If (v^j_{l+1}, v^j_{l+2}) is a right edge then let v^j_f be the first node after v^j_l in S such that (v^j_f, v^j_{f+1}) is a left edge.

- If (v^j_k, v^j_{k+1}) is a left edge, then let v^j_l be the first node after v^j_k in S such that (v^j_l, v^j_{l+1}) is a right edge.
  - If (v^j_{l+1}, v^j_{l+2}) is a right edge then let v^j_f be the first node after v^j_l in S such that (v^j_f, v^j_{f+1}) is a left edge.
  - If (v^j_{l+1}, v^j_{l+2}) is a left edge then let v^j_f be the first node after v^j_l in S such that (v^j_f, v^j_{f+1}) is a right edge.

Notice that S^q_j could have no vertex v^q_l or v^q_f if the spine ends before a left edge or a right edge is encountered. Roughly speaking, a path S_j starts from the vertex v^j_k of S following the one where path S_{j-1} ends; further, S_j ends where the second or the third alternation among left and right edges of S is found starting from v^j_k. In particular, S_j ends at the third alternation when the second alternation comes immediately after the first one, i.e., when S contains a sequence (left edge, right edge, left edge) or a sequence (right edge, left edge, right edge) providing the first and the second alternation among left and right edges of S starting from v^j_k.

Tree T is also subdivided into subtrees: For $0 \leq j \leq q$, tree T_j is the subtree of T induced by the nodes in S_j and the nodes in the subtrees rooted at non-spine nodes children of spine nodes in S_j.

Now we show how to construct a drawing Γ_j of each T_j, for $0 \leq j \leq q$. First, we show how to draw path S_j together with some other nodes of T_j. We distinguish eight cases, based on whether:

- j is even (Cases 1-2-3-4) or odd (Cases 5-6-7-8)
- (v^j_k, v^j_{k+1}) is a right edge (Cases 1-2-5-6) or a left edge (Cases 3-4-7-8)
- (v^j_{l+1}, v^j_{l+2}) is a right edge (Cases 1-3-5-7) or a left edge (Cases 2-4-6-8)

In the cases in which j is even, draw $L(T_j)$ and $R(T_j)$ on l_{2d+1} so that the each node of $L(T_j)$ (of $R(T_j)$) is one unit to the right of its left child (resp. one unit to the left of its right child). In the cases in which j is odd, draw $L(T_j)$ and $R(T_j)$ on u_{2d+1} so that the each node of $L(T_j)$ (of $R(T_j)$) is one unit to the left of its left child (resp. one unit to the right of its right child). Denote by h_l the vertical grid line passing through v_l, and denote by h_{l+1}, h_{l+2}, h_{l-1}, and h_{l-2} the vertical grid lines one unit to the right, two units to the right, one unit to the left, and two units to the left of h_l, respectively.

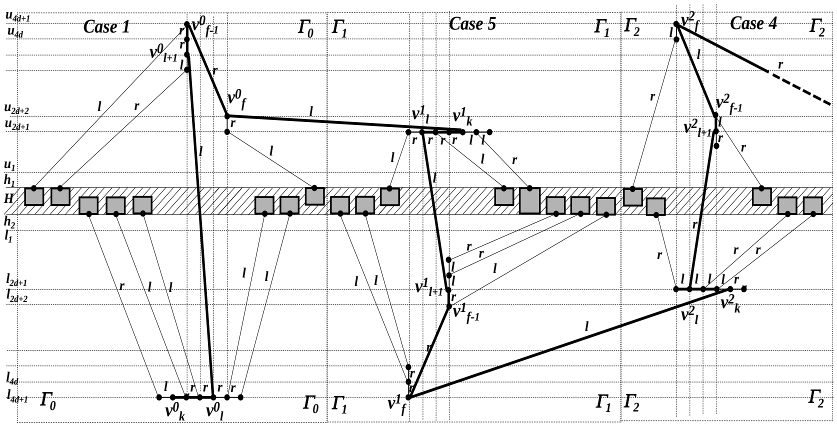


Figure 5.17: Thick segments represent the edges of S. Left (right) edges are labeled by l (resp. by r).

Case 1. Draw v^j_f at the intersection between h_{l+1} and u_{2d+2}; draw $R(T(v^j_f))$ on h_{l+1}, with any node one unit above its right child. Draw v^j_{f-1} at the intersection between h_{l-2} and u_{4d+1}. Draw $R(T(v^j_{l+1}))$ till v^j_{f-1} on h_{l-2}, with any node one unit below its right child; draw $L(T(v^j_{l+1}))$ on h_{l-2}, with any node one unit above its left child.

Case 2. Draw v^j_f at the intersection between h_{l-2} and u_{4d+1}; draw $L(T(v^j_f))$ on h_{l-2}, with any node one unit above its left child. Draw v^j_{f-1} at the intersection between h_{l+1} and u_{2d+2}. Draw $L(T(v^j_{l+1}))$ till v^j_{f-1} on h_{l+1}, with any node one unit below its left child; draw $R(T(v^j_{l+1}))$ on h_{l+1}, with any node one unit above its right child.

Case 3. Draw v^j_f at the intersection between h_{l+2} and u_{2d+2}; draw $R(T(v^j_f))$ on h_{l+2}, with any node one unit above its right child. Draw v^j_{f-1} at the intersection between h_{l-1} and u_{4d+1}. Draw $R(T(v^j_{l+1}))$ till v^j_{f-1} on h_{l-1}, with any node one unit below its right child; draw $L(T(v^j_{l+1}))$ on h_{l-1}, with any node one unit above its left child.

Case 4. Draw v^j_f at the intersection between h_{l-1} and u_{4d+1}; draw $L(T(v^j_f))$

on h_{l-1}, with any node one unit above its left child. Draw v^j_{f-1} at the intersection between h_{l+2} and u_{2d+2}. Draw $L(T(v^j_{l+1}))$ till v^j_{f-1} on h_{l+2}, with any node one unit below its left child; draw $R(T(v^j_{l+1}))$ on h_{l+2}, with any node one unit above its right child.

Case 5. Draw v^j_f at the intersection between h_{l-1} and l_{4d+1}; draw $R(T(v^j_f))$ on h_{l-1}, with any node one unit below its right child. Draw v^j_{f-1} at the intersection between h_{l+2} and l_{2d+2}. Draw $R(T(v^j_{l+1}))$ on h_{l+2}, with any node one unit above its right child; draw $L(T(v^j_{l+1}))$ on h_{l+2}, with any node one unit below its left child.

Case 6. Draw v^j_f at the intersection between h_{l+2} and l_{2d+2}; draw $L(T(v^j_f))$ on h_{l+2}, with any node one unit below its left child. Draw v^j_{f-1} at the intersection between h_{l-1} and l_{4d+1}. Draw $L(T(v^j_{l+1}))$ on h_{l-1}, with any node one unit above its left child; draw $R(T(v^j_{l+1}))$ on h_{l-1}, with any node one unit below its right child.

Case 7. Draw v^j_f at the intersection between h_{l-2} and l_{4d+1}; draw $R(T(v^j_f))$ on h_{l-2}, with any node one unit below its right child. Draw v^j_{f-1} at the intersection between h_{l+1} and u_{2d+2}. Draw $R(T(v^j_{l+1}))$ on h_{l+1}, with any node one unit above its right child; draw $L(T(v^j_{l+1}))$ on h_{l+1}, with any node one unit below its left child.

Case 8. Draw v^j_f at the intersection between h_{l+1} and l_{2d+2}; draw $L(T(v^j_f))$ on h_{l+1}, with any node one unit below its left child. Draw v^j_{f-1} at the intersection between h_{l-2} and l_{4d+1}. Draw $L(T(v^j_{l+1}))$ on h_{l-2}, with any node one unit above its left child; draw $R(T(v^j_{l+1}))$ on h_{l-2}, with any node one unit below its right child.

In Γ_0 shift the nodes of $L(T_0)$ and $R(T_0)$ vertically, so that they keep the same x-coordinates and lie on line l_{4d+1}.

Fig. 5.17 shows how to draw path S_j in Cases 1, 5, and 4.

For each T_j, with $0 \leq j \leq q$, recursively construct a drawing of each subtree rooted at a node of T_j that has not been already drawn and that is child of a node of T_j that has been already drawn. Let h_{max} be the maximum between the heights of the drawings of the subtrees recursively drawn. Set the distance between h_1 and h_2 to be $h_{max} - 1$, that is H consists of h_{max} horizontal grid lines.

For each T_j, with $0 \leq j \leq q$ and j even, construct a drawing Γ_j starting from the drawing of S_j already constructed, as follows:

- Consider the drawings of the subtrees rooted at non-already drawn nodes of T_j that are children of nodes belonging to $L(T_j)$ or to $R(T_j)$, and whose parents are placed to the right of v^j_l. Place such drawings in the left-right order induced by the order of their parents on l_{4d+1} or on l_{2d+1}, at one unit of horizontal distance between them, with the leftmost vertical line intersecting the leftmost drawing one unit to the right of the last node of $R(T_j)$, and with their leftmost and rightmost paths on h_1.

- Consider the drawings of the subtrees rooted at non-already drawn nodes of T_j that are children of nodes belonging to $L(T_j)$ or to $R(T_j)$, and whose parents are placed to the left of v^j_l. Place such drawings in the left-right order induced by the order of their parents on l_{4d+1} or on l_{2d+1}, at one unit of horizontal distance between them, with the rightmost vertical line intersecting the rightmost drawing one unit to the left of the last node of $L(T_j)$, and with their leftmost and rightmost paths on h_1.

- Consider the drawings of the subtrees rooted at non-already drawn nodes of T_j that are children of nodes already drawn in the upper part of the drawing. Rotate such drawings of π radiants and place them so that their leftmost and rightmost paths lie on h_2, so that the drawings of the subtrees rooted at children of nodes drawn on h_{l+1} or on h_{l+2} (on h_{l-1} or on h_{l-2}) are placed to the right (resp. to the left) of the drawing constructed up to now, at one unit of horizontal distance in the order induced by their parents in $L(T(v^j_{l+1}))$ or in $R(T(v^j_{l+1}))$.

If j is odd, a drawing Γ_j of T_j can be constructed analogously.

Now place all the Γ_j's together, starting from Γ_0, and iteratively adding Γ_j, for $j = 1, \ldots, m$, so that the leftmost vertical line intersecting Γ_j is one unit to the right of the rightmost vertical line intersecting Γ_{j-1}.

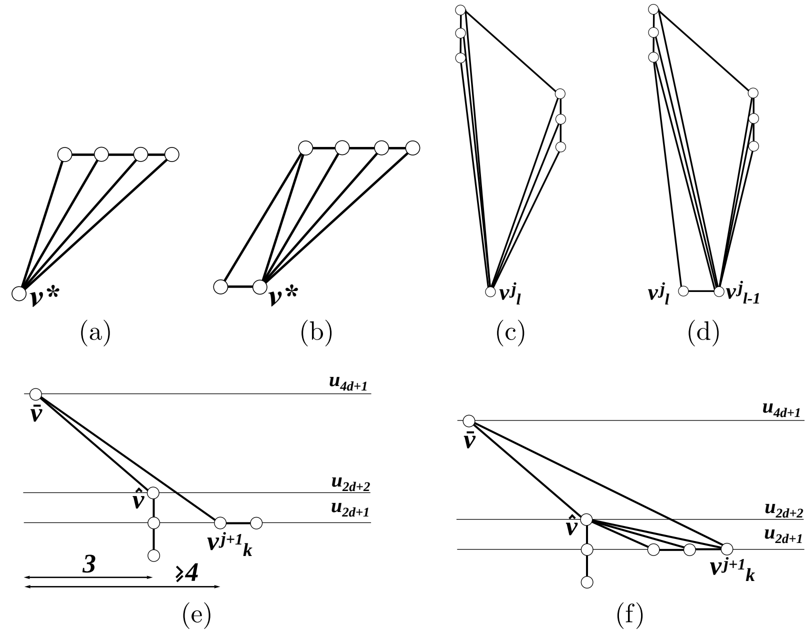
We prove that the resulting drawing Γ is a star-shaped drawing of T.

Lemma 5.11 Γ *is a star-shaped drawing of* T*.*

Proof: We show that, given a rooted binary tree T, dual of a maximal outerplanar graph G, the above described algorithm constructs a star-shaped drawing Γ of T.

First, observe that the claim that all the subtrees recursively drawn are contained inside H holds, since the distance between h_1 and h_2 is set equal to the height of the highest subtree recursively drawn.

Second, all the nodes directly drawn are in the upper and lower part of Γ. Namely, we have that: (i) by construction such nodes are never placed above u_{4d+1} or below l_{4d+1} and (ii) if such nodes are placed in H, then it's easy to deduce by the algorithm's construction that there exists a leftmost or a rightmost path of a subtree of T whose length is greater than d; however, this would imply that there exists a vertex of G whose degree is greater than d, since all the nodes of a leftmost or rightmost path of a subtree of T are neighbors of the same node of G.



(a) (b) (c) (d)

(e) (f)

Figure 5.18: Illustrations for the proof of Lemma 5.11.

Drawing Γ is straight-line and order-preserving by construction. We prove that Γ is planar and satisfies Conditions 1 and 2 of a star-shaped drawing. However, the planarity of Γ comes straightforwardly by the algorithm's construction, with the exception of the proof that each edge (v^j_f, v^{j+1}_k), with $1 \leq j \leq q-1$, does not cross other edges.

For each node v^* lying in the upper part of the drawing, except for v^j_{l-1} and v^j_l, all the nodes of $P_l(v^*)$ and of $P_r(v^*)$ are placed either on h_1, or in the upper part of Γ. Analogously, for each node v^* lying in the lower part of the drawing, except for v^j_{l-1} and v^j_l, all the nodes of $P_l(v^*)$ and of $P_r(v^*)$ are placed either on h_2, or in the lower part of Γ. Hence, all the edges incident to nodes different from v^j_{l-1} and v^j_l and lying in the upper part and in the lower part of the drawing do not cut H. It follows that, provided a proof that the edges connecting v^j_{l-1} and v^j_l to the nodes on $P_l(v^j_{l-1})$ and $P_r(v^j_{l-1})$, and to the nodes on $P_l(v^j_l)$ and $P_r(v^j_l)$, respectively, do not cause crossings, the proof that Γ is planar and satisfies Conditions 1 and 2 of a star-shaped drawing can be done separately for each recursive step of the algorithm.

Consider a subtree T_j of T, as defined for the algorithm's description. For each node v^* of T_j different from v^j_{l-1}, v^j_l, v^j_{f-1}, and v^j_f, one between $P_l(v^*)$ and $P_r(v^*)$ has nodes placed either all on h_1 or all on h_2, but for v^* (see Fig. 5.18 (a)); the other one between $P_l(v^*)$ and $P_r(v^*)$ has one node placed on the same horizontal or vertical line of v^* and the other nodes either all on h_1 or all on h_2 (see Fig. 5.18 (b)). Hence, straight-lines can be drawn from v^* to the nodes of $P_l(v^*)$ and of $P_r(v^*)$ without creating crossings in Γ.

Concerning v^j_{l-1} (resp. v^j_l), one between $P_l(v^j_{l-1})$ and $P_r(v^j_{l-1})$ (resp. one between $P_l(v^j_l)$ and $P_r(v^j_l)$) has nodes all on h_1 or all on h_2, but for v^j_{l-1} (resp. but for v^j_{l-1} and its child that lies on the same horizontal line of v^j_{l-1}) and the other one between $P_l(v^j_{l-1})$ and $P_r(v^j_{l-1})$ (resp. between $P_l(v^j_l)$ and $P_r(v^j_l)$) is a convex polygon $\mathcal{P}$ (see Figs. 5.18.c and 5.18.d), cutting H. However, no recursively drawn subtree has intersection with the smallest vertical strip containing $\mathcal{P}$ (such a strip is delimited either by lines h_{l-2} and h_{l+1} or by lines h_{l-1} and h_{l+2}). Hence, straight-lines can be drawn from v^j_{l-1} (from v^j_l) to the nodes of $P_l(v^j_{l-1})$ and of $P_r(v^j_{l-1})$ (resp. of $P_l(v^j_l)$ and of $P_r(v^j_l)$) without creating crossings in Γ.

Finally, consider nodes v^j_{f-1} and v^j_f. One out of v^j_{f-1} and v^j_f, say $\overline{v}$, is placed on u_{4d+1} or on l_{4d+1}, while the other one, say $\hat{v}$, is placed on u_{2d+2} or on l_{2d+2}. One between $P_l(\overline{v})$ and $P_r(\overline{v})$ has nodes all on h_1 or all on h_2, but for $\overline{v}$, while the other one has nodes all on u_{2d+1} or on l_{2d+1}, but for $\overline{v}$ and , eventually, for $\hat{v}$. To prove that $\overline{v}$ is visible from the nodes of $P_l(\overline{v})$ and $P_r(\overline{v})$, we claim that the slope of the edge connecting $\overline{v}$ to v^{j+1}_k is less than the one of the edge connecting $\overline{v}$ to $\hat{v}$ (see Fig. 5.18 (e)). Namely, the horizontal distance between $\overline{v}$ and v^{j+1}_k is at least 4 and the vertical distance between $\overline{v}$ and v^{j+1}_k is exactly $2d$. Further, the horizontal distance between $\overline{v}$ and $\hat{v}$ is exactly 3 and the vertical distance between $\overline{v}$ and $\hat{v}$ is exactly $2d-1$, and so the slope of $(\overline{v}v^{j+1}_k)$ is less or equal than $\frac{2d}{4}$, while the one of $(\overline{v}\hat{v})$ is $\frac{2d-1}{3}$. We have that

$\frac{2d}{4} < \frac{2d-1}{3}$ if and only if $d > 2$ that is always satisfied considering maximal outerplanar graphs with more than 3 vertices. Hence, straight-lines can be drawn from $\overline{v}$ to the nodes of $P_l(\overline{v})$ and of $P_r(\overline{v})$ without creating crossings in Γ. Concerning $\hat{v}$, the nodes of one between $P_l(\hat{v})$ and $P_r(\hat{v})$ lie all on h_1, or all on h_2, but for $\hat{v}$ and eventually for its child that lies on the same vertical line of $\hat{v}$. The nodes of the other one between $P_l(\hat{v})$ and $P_r(\hat{v})$ lie all on u_{2d+1} or all on l_{2d+1}, but for $\hat{v}$ and eventually for $\overline{v}$, that we have already proved to be visible from $\hat{v}$. Hence, straight-lines can be drawn from $\hat{v}$ to the nodes of $P_l(\hat{v})$ and of $P_r(\hat{v})$ without creating crossings in Γ (see Fig. 5.18 (f)).

Concerning Condition 3, $L(T)$ and $R(T)$ lie on the bottommost line l_{4d+1} intersecting Γ. Hence, placing the poles u_l and u_r of G one unit below l_{4d+1} allows to draw edges from u and u_r to the nodes of $L(T)$ and $R(T)$ without creating crossings with Γ. □

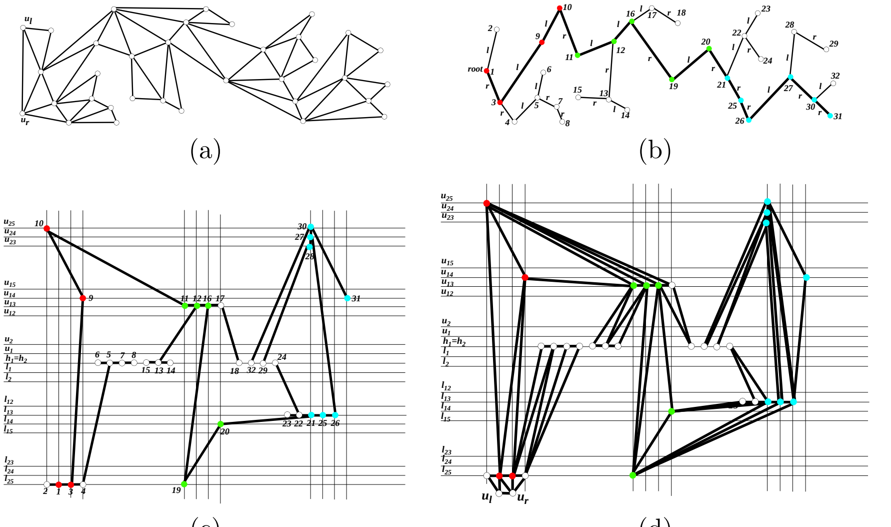
Let's analyze the area requirement of Γ. The width of Γ is clearly $O(n)$. The height of Γ is the sum of the heights of H, of the upper part, and of the lower part of Γ. The height of H is equal to the height of the highest subtree of T recursively drawn; by definition of S, each subtree recursively drawn has at most $n/2$ nodes. Denoting by $H(n)$ the maximum height of a drawing of an n-nodes tree T constructed by the algorithm, we get: $H(n) \leq (4d+1) + (4d+1) + H(n/2) = O(d) + H(n/2) = O(d \log n)$.

Place the poles of G' on the horizontal line one unit below v_0, at one horizontal unit distance from each other, and so that one pole is on the same vertical line of v_0. Notice that this placement doesn't asymptotically increase the area of Γ. Finally, the edges necessary to augment Γ in a drawing of G' can be inserted and the dummy edges inserted in the first step can be removed obtaining a drawing of G. Figure 5.19 shows an example of application of the above described algorithm.

Theorem 5.6 *Any n-vertex outerplanar graph of degree d has a straight-line outerplanar drawing in $O(dn \log n)$ area.*

Straightforwardly, we obtain the following:

Corollary 5.1 *Any n-vertex outerplanar graph with constant degree has a straight-line outerplanar drawing in $O(n \log n)$ area.*



(a)

(b)

(c)

(d)

Figure 5.19: (a) A maximal outerplanar graph G with $n = 34$ vertices and degree $d = 6$. The poles of G are labeled by u_l and u_r. (b) The dual binary tree T of G. Edges labeled by l (by r) are between a node and its left (resp. right) child. Thick edges show the spine S. Red, green, and blue vertices show subpaths S_0, S_1, and S_2 of S, respectively. Cases 2, 8, and 1 have to be applied to draw S_0, S_1, and S_2, respectively. (c) The star shaped drawing Γ of T constructed by the algorithm shown in Sect. 5.6. (d) The outerplanar drawing of G obtained by augmenting Γ with the poles of G and with extra edges.

5.7 Conclusions and Open Problems

In this chapter we have presented four linear-time algorithms for constructing planar straight-line grid drawings of outerplanar graphs. The first algorithm is for balanced outerplanar graphs; it constructs non-outerplanar drawings in $O(n)$ area and with angular resolution greater than $\frac{c}{\sqrt{n}}$, where c is a constant. The second algorithm is also for balanced outerplanar graphs; it constructs outerplanar drawings in $O(n)$ area and with angular resolution less than $\frac{c}{n}$, where c is a constant. The third algorithm is for general outerplanar graphs and constructs outerplanar drawings in $O(n^{1.48})$ area. The fourth algorithm is for general outerplanar graphs and constructs outerplanar drawings in $O(dn \log n)$ area. We have also shown that the outerplanar drawings of outerplanar graphs are strongly related to a special kind of drawings of their dual trees. We have called such drawings star-shaped drawings. This relationship leads to a new methodology for drawing outerplanar graphs and this can be helpful in the development of new algorithms, since drawing a tree, even if with some constraints, is generally easier than drawing a graph.

Despite of the improved upper bounds proved in this chapter, the following problems still remain widely open:

Open Problem 5.1 *Which are the asymptotic bounds for the area requirements of straight-line and poly-line planar drawings of outerplanar graphs?*

In fact, while the research efforts on the determination of the area requirements of straight-line and poly-line drawings of outerplanar graphs have produced many algorithms (and consequently some upper bounds), no lower bound better than the trivial $\Omega(n)$ is known. In [Bie02] Biedl conjectured an $\Omega(n \log n)$ lower bound on the area requirement of straight-line and poly-line drawings of outerplanar graphs. More precisely, she exhibited a class of outerplanar graphs, the "snowflake graphs" shown in Fig. 5.20 (a), for which she claimed:

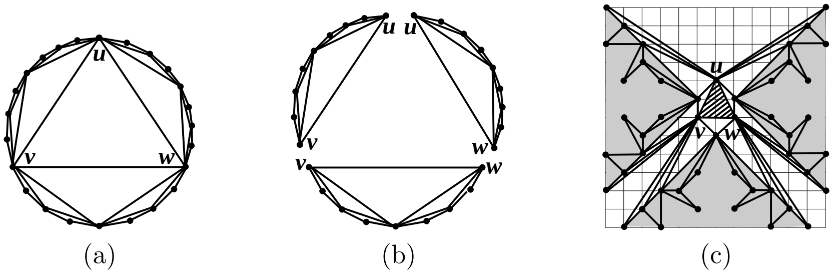


Figure 5.20: (a) A snowflake graph. (b) A snowflake graph subdivided into three complete outerplanar graphs. (c) Drawing snowflake graphs in linear area. The shaded regions correspond to the three copies of the drawing of the internal subgraph of a complete outerplanar graph constructed by the algorithm in Sect. 5.4.

Conjecture 5.1 *(Biedl [Bie02]) Any poly-line drawing of the snowflake graph has $\Omega(n \log n)$ area.*

It can be observed that an n-vertex snowflake graph G is composed of three identical $O(n)$-vertex complete outerplanar graphs with height h. Then, the algorithm shown in Sect. 5.4 can be used as follows. Construct a drawing of the internal subgraph $I(G)$ of a complete outerplanar graph with height $h' \geq h$ even (such an height can be always chosen to be at most one unit more than h), by means of the algorithm shown in Sect. 5.4. Such an algorithm constructs drawings in which the vertices of G_I that are connected to the poles lie on two half-lines l_1 and l_2 with a common endpoint and with slopes $-\pi/4$ and $\pi/4$. The entire drawing of $I(G)$ lies in the wedge with angle $\pi/2$ delimited by l_1 and l_2. Hence, considering three copies of the drawing of $I(G)$ constructed by the algorithm in Sect. 5.4, rotating them of 0, π, and $3\pi/2$, respectively, placing the drawings together, and inserting the three vertices that are poles for the three complete outerplanar graphs, an $O(\sqrt{n}) \times O(\sqrt{n})$ area straight-line drawing of a snowflake graph can be obtained (see Fig. 5.20 (c)).

We would like to point up that all the known algorithms for constructing straight-line drawings of general outerplanar graphs try to minimize the extension of the drawing in only one coordinate dimension, while allowing the other dimension to be $O(n)$. However, we believe that $O(n \log n)$ area cannot be achieved by squeezing the drawing in only one dimension, and that hence a compaction in both dimensions (or a proof that there exist n-vertex outerplanar graphs for which one dimension is $\Omega(n)$) should be pursued.

Conjecture 5.2 *There exist n-vertex outerplanar graphs for which, for any straight-line drawing in which the longest side of the bounding-box is $O(n)$, the smallest side of the bounding-box is $\omega(\log n)$.*

We notice that there exist outerplanar graphs such that both the width and the height of any grid drawing are $\Omega(\log n)$ (e.g., outerplanar graphs containing a complete ternary tree as a subgraph).

On the other hand, it is not clear whether a compaction in both dimensions can be pursued. In fact, no algorithm is known to construct planar grid drawings of outerplanar graphs in which both sides of the drawing are $o(n)$. However, our new techniques have not improved the situation in this direction.

Part III

Trees

Chapter 6

Straight-line, Poly-line, and Orthogonal Drawings of Trees

In this chapter[1] we consider straight-line, poly-line, orthogonal, and straight-line orthogonal drawings of trees in small area. After having exposed the state of the art on the area requirements of such drawings, we focus on straight-line orthogonal drawings of binary and ternary trees. We show algorithms for constructing order-preserving straight-line orthogonal drawings of binary trees in $O(n^{1.5})$ area, straight-line orthogonal drawings of ternary trees in $O(n^{1.631})$ area, and straight-line orthogonal drawings of complete ternary trees in $O(n^{1.262})$ area. Further, we show lower bounds on the area requirements and on the side lengths of straight-line orthogonal drawings of binary and ternary trees.

6.1 Introduction

The problem of drawing trees is a classical topic of investigation in algorithmics. Contributions on that field span almost three decades and algorithms for drawing trees have been proposed within a wide spectrum of drawing conventions. Here, we focus on straight-line, poly-line, orthogonal, and straight-line orthogonal drawings (addressing, for each of the drawing conventions, also the upward and the order-preserving constraint).

Straight-line Drawings. The best bound for constructing general trees is, as far as we know, the $O(n \log n)$ area upper bound provided by a sim-

[1] The contents of this chapter appeared in [Fra07c].

ple modification of the *hv*-drawing algorithm of Crescenzi, Di Battista, and Piperno [CBP92]. See Fig. 6.1 for an illustration of such an algorithm. On the other hand, no super-linear lower bound is known. Linear area bounds for straight-line drawings of trees have been however achieved when restricting to trees of bounded degree. In fact, Garg and Rusu presented an algorithm to construct straight-line drawings of binary trees in $O(n)$ area [GR02, GR04] and an algorithm to construct straight-line drawings of trees whose degree is $O(\sqrt{n})$ in $O(n)$ area [GR03c].

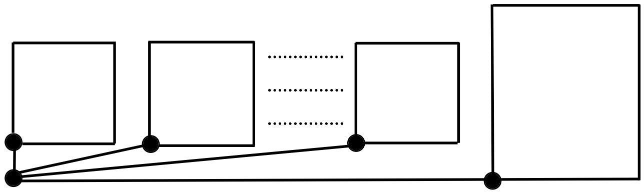


Figure 6.1: The *hv*-drawing algorithm of [CBP92] extended to work a general tree T. The heaviest subtree of the root $r(T)$ is the only subtree of $r(T)$ which has its root placed on the same horizontal line of $r(T)$.

Concerning *order-preserving drawings*, Garg and Rusu have shown in [GR03b] how to construct order-preserving straight-line drawings of general trees in $O(n \log n)$ area. A better bound of $O(n \log \log n)$ area can be achieved when restricting to binary trees [GR03b]. Both results rely on inductive algorithms exploiting interesting decompositions of trees into paths and nice geometric constructions of the final drawing from the drawings of the subtrees. No super-linear area lower bound is known, neither for binary nor for general trees.

Concerning *upward drawings*, the illustrated algorithm of Crescenzi *et al.* [CBP92] achieves the best upper bound of $O(n \log n)$. For trees with constant degree, Shin, Kim, and Chwa prove in [SKC00] that upward straight-line drawings in $O(n \log \log n)$ area can be constructed. No super-linear area lower bound is known, neither for binary nor for general trees.

Concerning *strictly-upward drawings*, tight bounds are known. In fact, the algorithm of Crescenzi *et al.* [CBP92] can be suitably modified in order to obtain strictly-upward drawings (in fact, instead of aligning the subtrees of the root with their bottom sides on the same horizontal line, it is sufficient to align them with their left sides on the same vertical line). The same authors also showed a binary tree requiring $\Omega(n \log n)$ area in any strictly-upward drawing, hence their bound is tight.

Concerning *strictly-upward order-preserving drawings*, the $\Omega(n \log n)$ area lower bound of Crescenzi *et al.* [CBP92] is still matched by an $O(n \log n)$ area upper bound in the case of binary trees. This was proved by Garg and Rusu in [GR03b]. However, for general trees, the best upper bound we are aware of is a $O(n4^{\sqrt{2 \log n}})$ due to Chan [Cha02].

Table 6.1 summarizes the best known area bounds for straight-line drawings of trees.

	Ord. Pres.	*Upw.*	*Str. Upw.*	*Upper Bound*	*Ref.*	*Lower Bound*	*Ref.*
Binary				$O(n)$	[GR04]	$\Omega(n)$	*trivial*
Binary	✓			$O(n \log \log n)$	[GR03b]	$\Omega(n)$	*trivial*
Binary		✓		$O(n \log \log n)$	[SKC00]	$\Omega(n)$	*trivial*
Binary			✓	$O(n \log n)$	[CBP92]	$\Omega(n \log n)$	[CBP92]
Binary	✓		✓	$O(n \log n)$	[GR03b]	$\Omega(n \log n)$	[CBP92]
General				$O(n \log n)$	[CBP92]	$\Omega(n)$	*trivial*
General	✓			$O(n \log n)$	[GR03b]	$\Omega(n)$	*trivial*
General		✓		$O(n \log n)$	[CBP92]	$\Omega(n)$	*trivial*
General			✓	$O(n \log n)$	[CBP92]	$\Omega(n \log n)$	[CBP92]
General	✓		✓	$O(n4^{\sqrt{2 \log n}})$	[Cha02]	$\Omega(n \log n)$	[CBP92]

Table 6.1: Summary of the best known area bounds for straight-line drawings of trees.

Poly-line Drawings. For general trees, no algorithms are known exploiting the possibility of bending the edges of the graph to get bounds better than the corresponding ones shown for straight-line drawings. However, better bounds can be achieved when restricting to bounded-degree trees.

Concerning *upward drawings*, a linear area bound is known, due to Garg *et al.* [GGT96], for all trees whose degree is $O(n^\delta)$, where δ is a constant less than 1. Concerning *order-preserving strictly-upward drawings*, Garg *et al.* also show how to construct order-preserving strictly-upward poly-line drawings of bounded degree trees in $O(n \log n)$ area. This bound is tight. Namely, there exist binary trees requiring $\Omega(n \log n)$ area in any (even non-strictly) upward order-preserving drawing [GGT96] (notice that for strictly-upward drawings the lower bound to be considered is still the one of Crescenzi *et al.* [CBP92]).

Table 6.1 summarizes the best known area bounds for poly-line drawings of trees.

Orthogonal Drawings. First, observe that orthogonal drawings only make sense for trees whose degree is at most four. Valiant proved in [Val81] that every n-node ternary tree (and every n-node binary tree) admits a $\Theta(n)$ area orthogonal drawing.

	Ord. Pres.	*Upw.*	*Str. Upw.*	*Upper Bound*	*Ref.*	*Lower Bound*	*Ref.*
Binary				$O(n)$	[GGT96]	$\Omega(n)$	*trivial*
Binary	✓			$O(n \log \log n)$	[GR03b]	$\Omega(n)$	*trivial*
Binary		✓		$O(n)$	[GGT96]	$\Omega(n)$	*trivial*
Binary			✓	$O(n \log n)$	[CBP92]	$\Omega(n \log n)$	[CBP92]
Binary	✓		✓	$O(n \log n)$	[GGT96]	$\Omega(n \log n)$	[CBP92]
General				$O(n \log n)$	[CBP92]	$\Omega(n)$	*trivial*
General	✓			$O(n \log n)$	[GR03b]	$\Omega(n)$	*trivial*
General		✓		$O(n \log n)$	[CBP92]	$\Omega(n)$	*trivial*
General			✓	$O(n \log n)$	[CBP92]	$\Omega(n \log n)$	[CBP92]
General	✓		✓	$O(n4^{\sqrt{2 \log n}})$	[Cha02]	$\Omega(n \log n)$	[CBP92]

Table 6.2: Summary of the best known area bounds for poly-line drawings of trees.

Concerning *order-preserving drawings*, in [DT81] Dolev and Trickey strengthened the result of Valiant, by proving that ternary trees (and binary trees) admit $\Theta(n)$ area order-preserving orthogonal drawings.

Concerning *upward drawings*, a $O(n \log \log n)$ bound for upward orthogonal drawings of binary trees was proved by Garg *et al.* in [GGT96]. The same authors proved the bound to be tight. In [Kim95] Kim showed that $O(n \log n)$ area is an optimal bound for upward orthogonal drawings of ternary trees, and that the bound is tight.

Concerning *order-preserving upward drawings*, $\Theta(n \log n)$ is an optimal bound both for binary and ternary trees. In fact, Kim [Kim95] proved the upper bound for ternary trees, result that can be immediately be extended to binary trees. The lower bound directly comes from the results of Garg *et al.* on order-preserving upward (non-orthogonal) drawings [GGT96].

Table 6.1 summarizes the best known area bounds for orthogonal drawings of trees.

	Ord. Pres.	*Upw.*	*Upper Bound*	*Ref.*	*Lower Bound*	*Ref.*
Binary			$O(n)$	[Val81]	$\Omega(n)$	*trivial*
Binary	✓		$O(n)$	[DT81]	$\Omega(n)$	*trivial*
Binary		✓	$O(n \log \log n)$	[GGT96]	$\Omega(n \log \log n)$	[GGT96]
Binary	✓	✓	$O(n \log n)$	[Kim95]	$\Omega(n \log n)$	[GGT96]
Ternary			$O(n)$	[Val81]	$\Omega(n)$	*trivial*
Ternary	✓		$O(n)$	[DT81]	$\Omega(n)$	*trivial*
Ternary		✓	$O(n \log n)$	[Kim95]	$\Omega(n \log n)$	[GGT96]
Ternary	✓	✓	$O(n \log n)$	[Kim95]	$\Omega(n \log n)$	[GGT96]

Table 6.3: Summary of the best known area bounds for orthogonal drawings of binary and ternary trees.

Straight-line Orthogonal Drawings. Straight-line orthogonal drawings provide extremely high readability of the combinatorial structure of a tree. In fact, the effects of orthogonal drawings (i.e., large angles between any two adjacent edges) and the ones of straight-line drawings (i.e., edges that extend straight) are combined in such a drawing convention. See Fig. 6.2 for an example of straight-line orthogonal drawing.

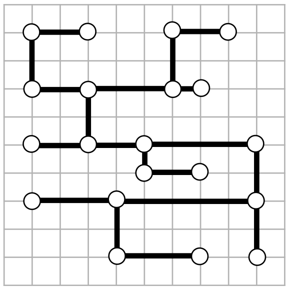


Figure 6.2: A straight-line orthogonal drawing of a binary tree.

Concerning the area requirements of straight-line orthogonal drawings, Chan, Goodrich, Kosaraju, and Tamassia in [CGKT02], and Shin, Kim, and Chwa in [SKC00] have shown that $O(n \log\log n)$ area suffices for straight-line orthogonal drawings of binary trees. Further, it has been shown in [CBP92, CGKT02] that binary trees admit upward straight-line orthogonal drawings in $O(n \log n)$ area. Such an area bound is worst-case optimal, as proved in [CGKT02]. Crescenzi *et al.* [CBP92] proved that complete binary trees admit upward straight-line orthogonal drawings in linear area, improving the result of Shiloach [Shi76], stating that complete binary trees admit straight-line orthogonal drawings in linear area.

In this chapter we study straight-line orthogonal drawings of binary and ternary trees and we present the following results: (i) order-preserving straight-line orthogonal drawings of binary trees can be constructed in $O(n^{1.5})$ area; (ii) upward order-preserving straight-line orthogonal drawings of binary trees require (and can be realized in) $\Omega(n^2)$ area; (iii) straight-line orthogonal drawings of ternary trees can be constructed in $O(n^{1.631})$ area; (iv) order-preserving straight-line orthogonal drawings of ternary trees require (and can be realized in) $\Omega(n^2)$ area; (v) straight-line orthogonal drawings of complete ternary trees can be constructed in $O(n^{1.262})$ area; and (vi) there exist ternary trees for which the minimum side of any straight-line orthogonal drawing is $\Omega(n^{0.438})$

and, for complete ternary trees, such a bound is tight.

Table 6.4 summarizes the best known area bounds for straight-line orthogonal drawings of binary and ternary trees.

	Upw.	*Ord. Pres.*	*Upper Bound*	*Ref.*	*Lower Bound*	*Ref.*
Complete Binary	✓		$O(n)$	[CBP92]	$\Omega(n)$	trivial
Complete Binary			$O(n)$	[Shi76]	$\Omega(n)$	trivial
Binary	✓	✓	$O(n^2)$	[CBP92]	$\Omega(n^2)$	Th. 6.1
Binary	✓		$O(n \log n)$	[CBP92]	$\Omega(n \log n)$	[CGKT02]
Binary		✓	$O(n^{1.5})$	Th. 6.2	$\Omega(n)$	trivial
Binary			$O(n \log \log n)$	[CGKT02]	$\Omega(n)$	trivial
Complete Ternary	✓		*non-drawable*			
Complete Ternary			$O(n^{1.262})$	Th. 6.6	$\Omega(n)$	trivial
Ternary	✓	✓	*non-drawable*			
Ternary	✓		*non-drawable*			
Ternary		✓	$O(n^2)$	Th. 6.4	$\Omega(n^2)$	Th. 6.3
Ternary			$O(n^{1.631})$	Th. 6.5	$\Omega(n)$	trivial

Table 6.4: Summary of the best known area bounds for straight-line orthogonal drawings of binary and ternary trees. For complete trees the order-preserving column is not considered, since such trees are symmetric. Straight-line orthogonal upward drawings of ternary trees cannot generally be constructed.

The rest of the chapter is organized as follows. In Sect. 6.2 we study straight-line orthogonal drawings of binary trees; in Sect. 6.3 we study straight-line orthogonal drawings of ternary trees; in Sect. 6.4 we study straight-line orthogonal drawings of complete ternary trees; finally, in Sect. 6.5 we conclude and present some open problems.

6.2 Straight-Line Orthogonal Order-Preserving Drawings of Binary Trees

First, we show that order-preserving upward straight-line orthogonal drawings of binary trees generally require quadratic area. Such a bound is matched by an $O(n^2)$ upper bound obtained by using the well-known *h-v layout* (see, e.g., [CBP92]).

Theorem 6.1 *There exists an n-node binary tree T requiring $\Omega(n^2)$ area in any upward order-preserving straight-line orthogonal drawing.*

Proof: Assume $n \equiv 0 \bmod 6$. Tree T is composed of (see Fig. 6.3 (a)):

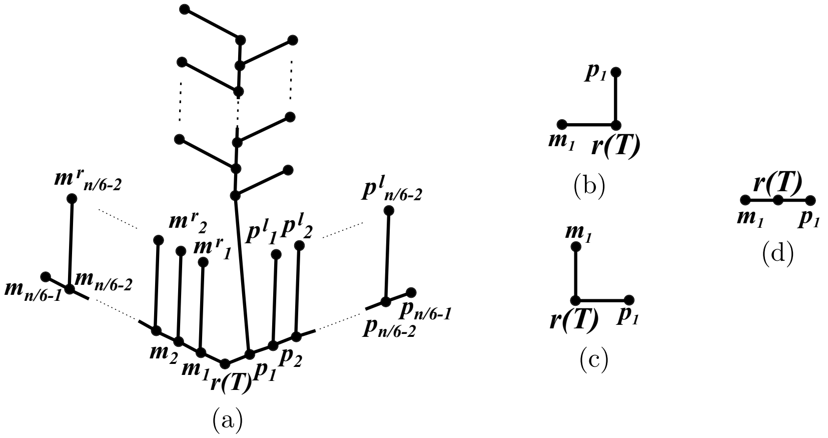


Figure 6.3: (a) Tree T providing the lower bound of Theorem 6.1. (b)-(c)-(d) Possible placements of $r(T)$ and its children.

- an $n/6$-node spine $C_1 : (m_0 = r(T), m_1, \dots, m_{\frac{n}{6}-2}, m_{\frac{n}{6}-1})$, with m_i left child of m_{i-1}, for $1 \leq i \leq \frac{n}{6} - 1$;
- an $n/6$-node spine $C_2 : (p_0 = r(T), p_1, \dots, p_{\frac{n}{6}-2}, p_{\frac{n}{6}-1})$, with p_i right child of p_{i-1}, for $1 \leq i \leq \frac{n}{6} - 1$;
- the right child m_i^r of each node m_i of C_1, with $1 \leq i \leq \frac{n}{6} - 2$;
- the left child p_i^l of each node p_i of C_2, with $1 \leq i \leq \frac{n}{6} - 2$;
- a path C_3 of $n/6 + 3$ nodes, alternating between right and left children, such that one end-vertex of C_3 is m_1^r; and
- $n/6 + 3$ leaves attached to C_3, alternating between left and right children.

Consider any upward order-preserving straight-line orthogonal drawing Γ of T. In [GGT96] it is shown that C_3 and its attached leaves require $\Omega(n)$ height in any upward order-preserving drawing.

Consider the relative position of $r(T)$ and its children in Γ. Three are the cases; either m_1 is to the left of $r(T)$ and p_1 is above $r(T)$ (see Fig. 6.3 (b)), or m_1 is above $r(T)$ and p_1 is to the right of $r(T)$ (see Fig. 6.3 (c)), or m_1 is to the left of $r(T)$ and p_1 is to the right of $r(T)$ (see Fig. 6.3 (d)).

Suppose m_1 is to the left of $r(T)$. We prove by induction that each node m_i of C_1, with $1 \leq i \leq \frac{n}{6} - 1$, is drawn at least one unit to the left of its parent. The claim holds in the base case by the assumption that m_1 is to the left of $m_0 = r(T)$. If m_i is to the left of m_{i-1}, then the edges from m_i to its children are drawn towards the left and the top. Since the drawing is order-preserving, m_i^r must be above m_i and m_{i+1} to the left of m_i. So each node m_i, with $1 \leq i \leq \frac{n}{6} - 1$, is drawn at least one unit to the left of its parent, implying a linear lower bound on the width of Γ.

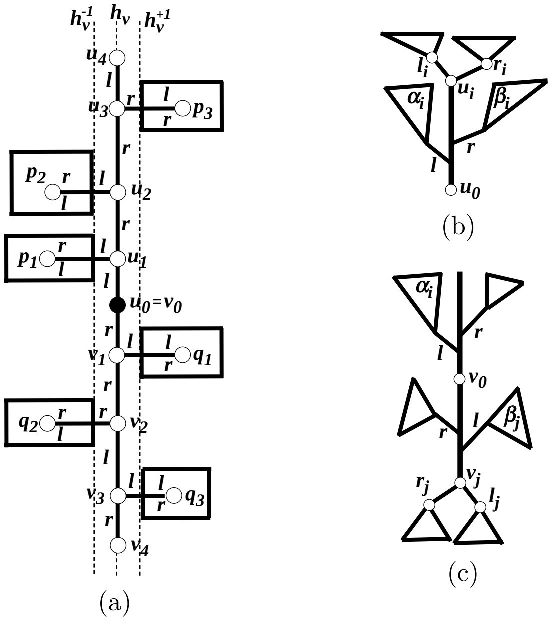
If m_1 is not to the left of $r(T)$ then p_1 is to the right of $r(T)$ and a similar argument shows that each node p_i, with $1 \leq i \leq \frac{n}{6} - 1$, is at least one unit to the right of its parent, again implying a linear lower bound on the width of Γ. Hence both the height and the width of Γ are $\Omega(n)$. □

Now we turn to non-upward drawings, showing that sub-quadratic area suffices for order-preserving straight-line orthogonal drawings:

Theorem 6.2 *Any n-node binary tree T admits an $O(n^{1.5})$ area order-preserving straight-line orthogonal drawing.*

Proof: We describe an inductive algorithm constructing an order-preserving straight-line orthogonal drawing Γ of T satisfying the *side visibility property*. Denoting by r the horizontal line through $r(T)$, then Γ has the side visibility property if no node, but for $r(T)$, is placed on r and no edge crosses r. If $n = 1$, then Γ is trivially constructed. Suppose $n > 1$. Select a double-spine $\pi = (u_k, u_{k-1}, \ldots, u_1, u_0 = r(T) = v_0, v_1, \ldots, v_m)$ in T. How to choose π is discussed later. Denote by p_i the non-spine child of a node $u_i \in \pi$ and by q_j the non-spine child of a node $v_j \in \pi$.

Recursively construct drawings $\Gamma(p_i)$ of $T(p_i)$ and $\Gamma(q_j)$ of $T(q_j)$ satisfying the side visibility property, for $1 \leq i < k$ and $1 \leq j < m$. Let h_v, h_v^{-1}, and h_v^1 be vertical grid lines with h_v^{-1} (h_v^1) one unit to the left (to the right) of h_v. Draw $r(T)$ on h_v. For $i = 1, 2, \ldots, k-1$, if p_i is the left child of u_i rotate $\Gamma(p_i)$ of π and place it so that the rightmost vertical line intersecting it is h_v^{-1} and with the lowest horizontal line intersecting it one unit above the highest horizontal line intersecting $\Gamma(p_{i-1})$ or u_{i-1}; otherwise (p_i is the right child of u_i), place $\Gamma(p_i)$ so that the leftmost vertical line intersecting it is h_v^1 and with the lowest horizontal line intersecting it one unit above the highest horizontal line intersecting $\Gamma(p_{i-1})$ or u_{i-1}. Draw u_i on h_v on the same horizontal line of its already drawn child (or one unit above the highest horizontal line intersecting $\Gamma(p_{i-1})$ or u_{i-1} if no child of u_i has been drawn). Draw u_k on h_v one unit above the highest horizontal line intersecting $\Gamma(p_{k-1})$ or u_{k-1}. For



(b)

(c)

(a)

Figure 6.4: Illustrations for the algorithm in the proof of Theorem 6.2. Left (right) edges are labeled l (r). Label l (r) inside a subtree shows the direction of the edge from the root to its left (right) child.

$j = 1, 2, \ldots, m-1$, if q_j is the right child of v_j rotate $\Gamma(q_j)$ of π and place it so that the rightmost vertical line intersecting it is h_v^{-1} and with the highest horizontal line intersecting it one unit below the lowest horizontal line intersecting $\Gamma(q_{j-1})$ or v_{j-1}; otherwise (q_j is the left child of v_j), place $\Gamma(q_j)$ so that the leftmost vertical line intersecting it is h_v^1 and with the highest horizontal line intersecting it one unit below the lowest horizontal line intersecting $\Gamma(q_{j-1})$ or v_{j-1}. Draw v_j on h_v on the same horizontal line of its already drawn child (or one unit below the lowest horizontal line intersecting $\Gamma(q_{j-1})$ or v_{j-1} if no child of v_j has been drawn). Draw v_m on h_v one unit below the lowest horizontal line intersecting $\Gamma(q_{m-1})$ or v_{m-1} (see Fig. 6.4 (a)).

It’s easy to see that the constructed drawing Γ is an order-preserving straight-line orthogonal drawing satisfying the side visibility property. Let’s analyze the area requirement of Γ. Concerning its height, there is at least one node of T on each horizontal grid line intersecting Γ, hence the height of Γ is

$O(n)$. Denote by $W(T)$ the width of the drawing constructed by the described algorithm when its input is binary tree T. Let also $W(n) = \max\{W(T)\}$ over all binary trees T with n nodes.

Since all subtrees drawn to the left (to the right) of π are aligned on their right side (on their left side) and since $W(n)$ is a non-decreasing function of n, then $W(n) = W(n_l) + W(n_r) + 1$, where n_l (n_r) is the number of nodes in the heaviest subtree drawn to the left (to the right) of π.

To get a good bound for $W(n)$ we need to carefully choose π. A technique similar to the one we present was introduced in [Cha02] for selecting (single) spines. π is composed of two spines $\mathcal{U} = (u_0, u_1, \ldots, u_k)$ and $\mathcal{V} = (v_0, v_1, \ldots, v_m)$. Spine $\mathcal{U}$ is iteratively selected as follows: $u_0 = r(T)$, u_1 is the left child of u_0. Denote by l_i and by r_i the left and right child of u_i, respectively. Denote also by α_i and by β_i the heaviest left subtree and the heaviest right subtree of path $(u_1, \ldots, u_{i-1})$ (see Fig. 6.4 (b)). If $|\alpha_i| + |T(r_i)| \leq |\beta_i| + |T(l_i)|$ then set $u_{i+1} = l_i$, otherwise set $u_{i+1} = r_i$. Spine $\mathcal{V}$ is iteratively selected as follows: $v_0 = r(T)$, v_1 is the right child of u_0. Denote by l_j and by r_j the left and right child of v_j, respectively. Denote by α_j the one between the heaviest right subtree of path $(v_1, \ldots, v_{j-1})$ and the heaviest left subtree of $\mathcal{U} \setminus u_0$ that has the greatest number of nodes. Denote also by β_j the one between the heaviest left subtree of path $(v_1, \ldots, v_{j-1})$ and the heaviest right subtree of $\mathcal{U} \setminus u_0$ that has the greatest number of nodes (see Fig. 6.4 (c)). If $|\alpha_j| + |T(l_j)| \leq |\beta_j| + |T(r_j)|$ then set $v_{j+1} = r_j$, otherwise set $v_{j+1} = l_j$. Similarly to [Cha02], we get the following:

Lemma 6.1 *For any left subtree α of $\mathcal{U} \setminus u_0$ or right subtree α of $\mathcal{V} \setminus v_0$ and for any right subtree β of $\mathcal{U} \setminus u_0$ or left subtree β of $\mathcal{V} \setminus v_0$, $|\alpha| + |\beta| \leq n/2$.*

Proof: If α and β are both subtrees of $\mathcal{U} \setminus u_0$ or if are both subtrees of $\mathcal{V} \setminus v_0$, then the statement follows as in Lemma 4.1 of [Cha02]. Otherwise, suppose α is a left subtree of $\mathcal{U} \setminus u_0$ and β is a left subtree of $\mathcal{V} \setminus v_0$. Let v_j be the parent of β's root. Denote by l_j and r_j the left and right child of v_j, respectively. Notice that $r_j = v_{j+1}$. Denote by α_j the one between the heaviest right subtree of path $(v_1, \ldots, v_{j-1})$ and the heaviest left subtree of $\mathcal{U} \setminus u_0$ that has the greatest number of nodes, and denote by β_j the one between the heaviest left subtree of path $(v_1, \ldots, v_{j-1})$ and the heaviest right subtree of $\mathcal{U} \setminus u_0$ that has the greatest number of nodes. By construction $|\alpha_j| + |T(l_j)| \leq |\beta_j| + |T(r_j)|$. Moreover, $|\alpha_j| + |T(l_j)| + |\beta_j| + |T(r_j)| \leq n$. Therefore, $\alpha_j + |T(l_j)| \leq n/2$. Since $\alpha \leq \alpha_j$ and $\beta = T(l_j)$, the statement follows. The case in which α is a right subtree of $\mathcal{V} \setminus v_0$ and β is a right subtree of $\mathcal{U} \setminus u_0$ is analogous. □

Selecting π as just described, we get $W(n) \leq \max_{n_1+n_2 \leq n/2} W(n_1) + W(n_2)+1$. As already noticed in [Cha02], by Hölder's inequality $n_1+n_2 \leq n/2$ implies $\sqrt{n_1}+\sqrt{n_2} \leq \sqrt{n}$ and, by induction, $W(n) \leq c\sqrt{n}-1$, for some constant c depending only on the values of $W(n)$ with n small. □

6.3 Straight-Line Orthogonal Drawings of Ternary Trees

In this section we consider straight-line orthogonal drawings of ternary trees. First, we show that if an order of the children of each node is fixed, then quadratic area is necessary in the worst case.

Theorem 6.3 *There exists an n-node ternary tree T requiring $\Omega(n^2)$ area in any order-preserving straight-line orthogonal drawing.*

Proof: Assume $n \equiv 4 \bmod 9$. Tree T is composed of (see Fig. 6.5 (a)):

- a spine $C_1 : (m_0 = r(T), m_1, \dots, m_{\frac{n-4}{9}}, m_{\frac{n+5}{9}})$, with m_1 left child of $r(T)$ and m_i middle child of m_{i-1}, for $i = 2, 3, \dots, \frac{n+5}{9}$;
- a spine $C_2 : (p_0 = r(T), p_1, \dots, p_{\frac{n-4}{9}}, p_{\frac{n+5}{9}})$, with p_i middle child of p_{i-1}, for $i = 1, 2, \dots, \frac{n+5}{9}$;
- a spine $C_3 : (q_0 = r(T), q_1, \dots, q_{\frac{n-4}{9}}, q_{\frac{n+5}{9}})$, with q_1 right child of $r(T)$ and q_i middle child of q_{i-1}, for $i = 2, 3, \dots, \frac{n+5}{9}$; and
- a left and a right child for each node m_i, p_i, and q_i, for $i = 1, 2, \dots, \frac{n-4}{9}$.

Consider any order-preserving straight-line orthogonal drawing of C_1 and of the children of nodes m_i, with $1 \leq i \leq \frac{n-4}{9}$. Suppose that m_1 is to the left of m_0. Then, to preserve the order of the children of m_1, m_i is to the left of m_{i-1}, for $i = 2, 3, \dots, \frac{n+5}{9}$. Analogously, if m_1 is to the right, above, or below m_0, then m_i is to the right, above, or below m_{i-1}, respectively, for $i = 2, 3, \dots, \frac{n+5}{9}$. Such an argument applies to C_2 (to C_3), as well: If p_1 (q_1) is to the left, to the right, above, or below p_0 (q_0), then p_i (q_i) is to the left, to the right, above, or below p_{i-1} (q_{i-1}), respectively, for $i = 2, 3, \dots, \frac{n+5}{9}$. Since $r(T)$ has three children, then at least one of them is above or below $r(T)$ and one of them is to the left or to the right of $r(T)$. Hence, any order-preserving straight-line orthogonal drawing of T has at least $\frac{n+5}{9}+1$ height and width. □

The proved bound is tight, as shown in the following:

Theorem 6.4 *Any n-node ternary tree T admits an $O(n^2)$ area order-preserving straight-line orthogonal drawing.*

Proof: We show an inductive algorithm constructing an order-preserving straight-line orthogonal drawing Γ of T satisfying the *top visibility property.* Denoting by l the vertical half-line starting at the root $r(T)$ of T and directed upward, then Γ has the top visibility property if no node, but for $r(T)$, is placed on l and no edge crosses l. If $n = 1$, then Γ is trivially constructed. Suppose $n > 1$. Let T_l, T_m, and T_r be the left, middle, and right subtree of $r(T)$. By induction, drawings Γ_l, Γ_m, and Γ_r satisfying the top visibility property can be constructed for T_l, T_m, and T_r, respectively. Draw $r(T)$ in the plane. Rotate Γ_l of $\pi/2$ in clockwise direction. Place Γ_l with the rightmost vertical line intersecting it one unit to the left of $r(T)$ and with $r(T_l)$ on the same horizontal line of $r(T)$. Rotate Γ_r of $\pi/2$ in counter-clockwise direction. Place Γ_r with the leftmost vertical line intersecting it one unit to the right of $r(T)$ and with $r(T_r)$ on the same horizontal line of $r(T)$. Place Γ_m with the highest horizontal line intersecting it one unit below the lowest horizontal line intersecting Γ_l or Γ_r and with $r(T_m)$ on the same vertical line of $r(T)$ (see Fig. 6.5 (b)). It's easy to see that Γ is an order-preserving straight-line orthogonal drawing satisfying the top visibility property. Since Γ has at least one node for each horizontal and vertical grid line intersecting it, then its height and its width are $O(n)$. □

For non-order-preserving drawings better bounds can be achieved:

Theorem 6.5 *Any n-node ternary tree T admits an $O(n^{1.631})$ area straight-line orthogonal drawing.*

Proof: We show an inductive algorithm that constructs a straight-line orthogonal drawing Γ of T satisfying the top visibility property (see the proof of Theorem 6.4). If $n = 1$, then Γ is trivially constructed. Suppose $n > 1$. Select a double-spine $\pi = (u_k, u_{k-1}, \dots, u_1, u_0 = r(T) = v_0, v_1, \dots, v_m)$ in T such that: $T(v_1)$ is the heaviest subtree of $r(T)$; for $j = 2, 3, \dots, m$, $T(v_j)$ is the heaviest subtree of v_{j-1}; $T(u_1)$ is the heaviest subtree of $r(T)$ different from $T(v_1)$; for $i = 2, 3, \dots, k$, $T(u_i)$ is the heaviest subtree of u_{i-1}. For each node v_j in π, with $j = 0, 1, \dots, m-1$, (for each node u_i in π, with $i = 1, 2, \dots, k-1$), call *second heaviest subtree* $S(v_j)$ ($S(u_i)$) and *third heaviest subtree* $R(v_j)$ ($R(u_i)$), the subtrees of v_j (of u_i) different from $T(v_{j+1})$ (from $T(u_{i+1})$) with the greater and the smaller number of nodes, respectively.

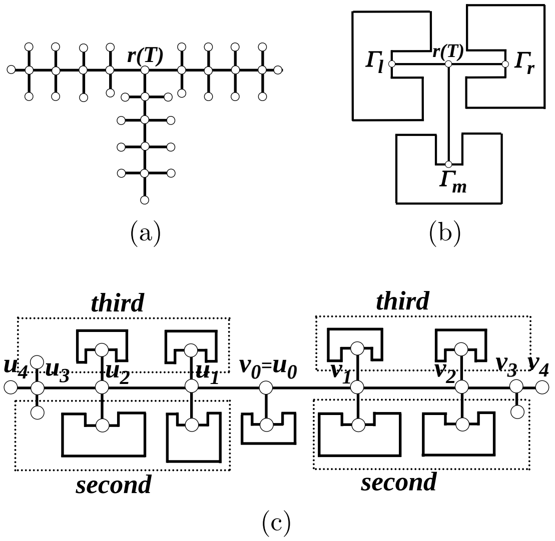


Figure 6.5: (a) Tree T requiring $\Omega(n^2)$ area in any order-preserving straight-line orthogonal drawing. (b) Illustration for the algorithm in Theorem 6.4. (c) Illustration for the algorithm in Theorem 6.5. Subtrees labeled by *second* or *third* are second or third heaviest subtrees, respectively.

Recursively construct a drawing satisfying the top visibility property of each subtree of π. Let h be an horizontal grid line. Draw $r(T)$ on h. Place the drawing of $R(v_0)$ with the highest horizontal line intersecting it one unit below h and with its root on the same vertical line of $r(T)$. For $j = 1, 2, \ldots, m-1$, rotate of π the drawing of $R(v_j)$. Place the drawing Γ_j^S of $S(v_j)$ and Γ_j^R of $R(v_j)$ so that the highest horizontal line intersecting Γ_j^S is one unit below h, the lowest horizontal line intersecting Γ_j^R is one unit above h, their roots are on the same vertical line, and the leftmost vertical line intersecting Γ_j^S or Γ_j^R is one unit to the right of the rightmost vertical line intersecting Γ_{j-1}^S, Γ_{j-1}^R, or v_{j-1}. Draw v_j on h on the same vertical line of its already drawn children (or draw v_j one unit to the right of the rightmost vertical line intersecting Γ_{j-1}^S, Γ_{j-1}^R, or v_{j-1} if no child of v_j has been drawn). Draw v_m on h one unit to the right of the rightmost vertical line intersecting Γ_{m-1}^S, Γ_{m-1}^R, or v_{m-1}. For path $(u_1, u_2, \ldots, u_k)$ and its subtrees, analogously construct a drawing in which the path lies on h, to the left of $r(T)$, and the $S(u_i)$'s and the $R(u_i)$'s are below and above h, respectively (see Fig. 6.5 (c)).

It's easy to see that Γ is a straight-line orthogonal drawing satisfying the top visibility property. Let's analyze the area of Γ. Since there is at least one node of T for each vertical grid line intersecting Γ, then its width is $O(n)$. Denote by $H(T)$ the height of the drawing constructed by the algorithm when its input is T. Let also $H(n) = \max\{H(T)\}$ over all ternary trees T with n nodes.

Since all subtrees drawn above π (below π) are aligned on their bottom side (on their top side) and since $H(n)$ is a non-decreasing function of n, then $H(n) = H(n_a)+H(n_b)+1$, where n_a (n_b) is the number of nodes in the heaviest subtree drawn above (below) of π. We claim (1) $n_a + n_b \leq 2n/3$. Namely, we have (2) $n_a \leq n_b$, (3) $n_b \leq n/2$, and (4) $n_a \leq n - 2n_b$.

Inequality (2) holds since for each node w in π, $|R(w)| \leq |S(w)|$; inequality (3) follows from the fact that, for each node v_j and u_i in π, $|S(v_j)| \leq |T(v_{j+1})|$ and $|S(u_i)| \leq |T(u_{i+1})|$; inequality (4) follows by considering any node v_j (u_i) in π and observing that $|S(v_j)| \leq |T(v_{j+1})|$ ($|S(u_i)| \leq |T(u_{i+1})|$) and that $|R(v_j)| + |S(v_j)| + |T(v_{j+1})| \leq n$ ($|R(u_i)| + |S(u_i)| + |T(u_{i+1})| \leq n$).

By (3) we have $n_b = \frac{n}{2} - \alpha$, with $\alpha \geq 0$. If $\alpha \geq n/6$, then by (2) $n_b + n_a \leq 2(n/2 - \alpha) \leq 2n/3$. If $\alpha < n/6$, by (4) we have $n_a \leq n - 2(n/2 - \alpha) = 2\alpha$. Hence, $n_b + n_a \leq n/2 - \alpha + 2\alpha = n/2 + \alpha \leq 2n/3$, and inequality (1) holds.

We claim that $n_a^{(1/\log_2 3)} + n_b^{(1/\log_2 3)} \leq n^{(1/\log_2 3)}$.

Hölder's inequality states that (5) $\sum a_i b_i \leq (\sum a_i^p)^{\frac{1}{p}} (\sum b_i^q)^{\frac{1}{q}}$ for every p and q such that $\frac{1}{p} + \frac{1}{q} = 1$. Substituting into (5) the values $a_1 = n_a^{(1/\log_2 3)}$, $a_2 = n_b^{(1/\log_2 3)}$, $b_1 = b_2 = 1$, $\frac{1}{p} = \frac{1}{\log_2 3}$, and $\frac{1}{q} = 1 - \frac{1}{\log_2 3}$, we get:

$$\begin{aligned}
n_a^{\frac{1}{\log_2 3}} + n_b^{\frac{1}{\log_2 3}} &\leq \left[\left(n_a^{\frac{1}{\log_2 3}}\right)^{\log_2 3} + \left(n_b^{\frac{1}{\log_2 3}}\right)^{\log_2 3}\right]^{\frac{1}{\log_2 3}} \cdot [1+1]^{\left(1-\frac{1}{\log_2 3}\right)} = \\
&= (n_a + n_b)^{\frac{1}{\log_2 3}} \cdot 2^{\left(1-\frac{1}{\log_2 3}\right)} \leq \left(\frac{2n}{3}\right)^{\frac{1}{\log_2 3}} \cdot 2^{\left(1-\frac{1}{\log_2 3}\right)} = \\
&= n^{\frac{1}{\log_2 3}} \left(2^{\frac{1}{\log_2 3}} \cdot 2 \cdot 2^{\frac{-1}{\log_2 3}}\right) / \left(3^{\frac{1}{\log_2 3}}\right) = \\
&= 2n^{\frac{1}{\log_2 3}} / 3^{\frac{1}{\log_2 3}} = n^{\frac{1}{\log_2 3}}.
\end{aligned}$$

Hence, $H(n) \leq \max_{n_1+n_2 \leq 2n/3} H(n_1) + H(n_2) + 1$ by induction solves to $H(n) = c \cdot n^{(1/\log_2 3)} - 1$ for some constant c, depending only on the values of

$H(n)$ with n small. It follows that $H(n) = O(n^{(1/\log_2 3)}) = O(n^{0.631})$ □

6.4 Straight-Line Orthogonal Drawings of Complete Ternary Trees

For complete ternary trees we present two algorithms constructing drawings with better area bounds than the ones obtained for general ternary trees.

Let Γ_h be a drawing of a complete ternary tree T_h with height h. In both algorithms inductively suppose to have a drawing Γ_{h-1} of T_{h-1} satisfying the top visibility property, take three copies Γ'_{h-1}, Γ''_{h-1}, and Γ'''_{h-1} of Γ_{h-1}, rotate Γ'_{h-1} of $\pi/2$ and Γ''_{h-1} of $3\pi/2$ in clockwise direction.

The algorithms differ in the geometric construction of Γ_h. In *Construction* 1 draw $r(T_h)$ on any horizontal grid line l_h. Place Γ'''_{h-1} with the highest horizontal line intersecting it one unit below l_h and with the root $r(T_{h-1})$ in Γ'''_{h-1} on the same vertical line of $r(T_h)$. Place Γ'_{h-1} with the rightmost vertical line intersecting it one unit to the left of the leftmost vertical line intersecting Γ'''_{h-1} and with the root $r(T_{h-1})$ in Γ'_{h-1} on l_h. Place Γ''_{h-1} with the leftmost vertical line intersecting it one unit to the right of the rightmost vertical line intersecting Γ'''_{h-1} and with the root $r(T_{h-1})$ in Γ''_{h-1} on l_h (see Fig. 6.6 (a)).

In *Construction* 2 draw $r(T_h)$ on any horizontal grid line l_h. Place Γ'_{h-1} with the rightmost vertical line intersecting it one unit to the left of $r(T_h)$ and with the root $r(T_{h-1})$ in Γ'_{h-1} on l_h. Place Γ''_{h-1} with the leftmost vertical line intersecting it one unit to the right of $r(T_h)$ and with the root $r(T_{h-1})$ in Γ''_{h-1} on l_h. Place Γ'''_{h-1} with the highest horizontal line intersecting it one unit below the lowest horizontal line intersecting Γ''_{h-1}, and with the root $r(T_{h-1})$ in Γ'''_{h-1} on the same vertical line of $r(T_h)$ (see Fig. 6.6 (b)). We have the following:

Theorem 6.6 *An n-node complete ternary tree T_h admits an $O(n^{1/\log_4 3}) = O(n^{1.262})$ area straight-line orthogonal drawing.*

Proof: Construct a drawing Γ_h of T_h by inductively using Construction 1. Denoting by W_h and by H_h the width and the height of Γ_h, respectively, by construction we have $W_h = W_{h-1} + 2H_{h-1}$ and $H_h = \max\{W_{h-1}, H_{h-1} + (W_{h-1} + 1)/2\}$. Assume by inductive hypothesis that $W_{h-1} = 2^{h-1} - 1$ and that $H_{h-1} = 2^{h-2}$. Notice that this holds in the base case, where $W_1 = H_1 = 1$. Observe also that by inductive hypothesis $H_{h-1} + (W_{h-1} + 1)/2 = 2^{h-2} + (2^{h-1} - 1 + 1)/2 = 2^{h-1} > W_{h-1} = 2^{h-1} - 1$. Hence, $H_h = 2^{h-1}$ and $W_h = 2^{h-1} - 1 + 2 \cdot 2^{h-2} = 2^h - 1$, that proves the inductive hypothesis. The area

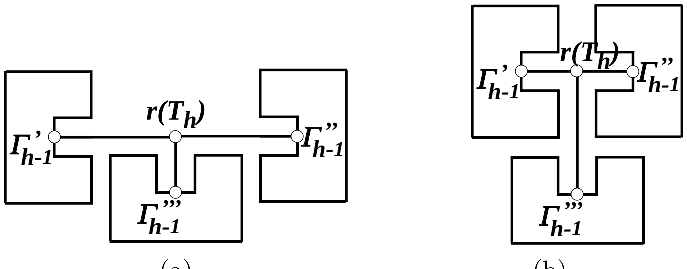


Figure 6.6: Constructions 1 (a) and 2 (b) for straight-line orthogonal drawings of complete ternary trees.

of Γ_h is equal to $(2^h - 1) \cdot 2^{h-1} < 4^h = 4^{O(\log_3 n)} = O(n^{1/\log_4 3})$. Inductively applying Construction 2 instead of Construction 1 yields to a drawing with asymptotically the same area. □

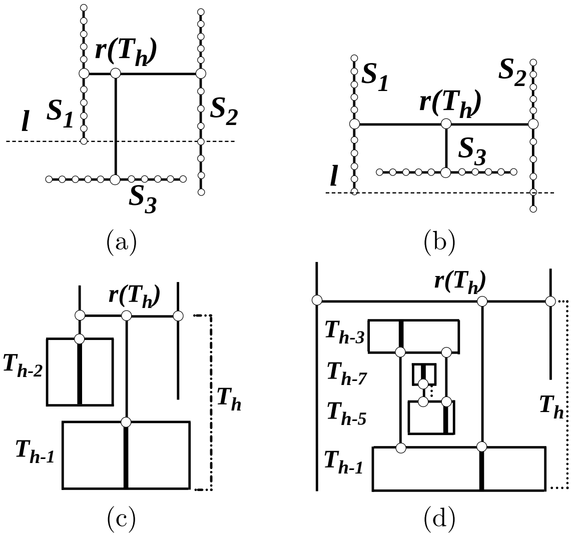
Next, we show that n-node complete ternary trees have $\Omega(n^{0.438})$ minimum side in any straight-line orthogonal drawing. This result sharply contrasts with the analogous for binary trees. Namely, *any* binary tree admits a straight-line orthogonal drawing in which one side is $O(\log n)$.

Let Γ_h be any straight-line orthogonal drawing of T_h. One of the children of $r(T_h)$, say v_1, is such that no other child of $r(T_h)$ is drawn on the line l through $r(T_h)$ and v_1. Moreover, for $i = 1, 2, \ldots, h-2$, node v_i has exactly one child v_{i+1} drawn on l. Hence, in any straight-line orthogonal drawing of T_h, there is a spine of h nodes drawn all on the same horizontal or vertical line l, such that no other child of $r(T_h)$ is on l. We call *leg* of Γ_h such a spine. Analogously, in any straight-line orthogonal drawing of T_h there is a double-spine of $2h-1$ nodes that are drawn all on the same horizontal or vertical line. We call *arm* of Γ_h such a double-spine. We have the following:

Lemma 6.2 *The minimum side of any straight-line orthogonal drawing of an n-node complete ternary tree is $\Omega(n^{0.438})$.*

Proof: Let Γ_h be a straight-line orthogonal drawing of a complete ternary tree T_h in which the length of the leg is minimum. Let $l(\Gamma_h)$ be the length of the leg in Γ_h. We claim that $l(\Gamma_h) \geq l(\Gamma_{h-1}) + l(\Gamma_{h-2})$.

Consider the arms of the subtrees of $r(T_h)$. Either two of such arms are vertical and one horizontal or vice versa. Assume, possibly rotating Γ_h of $\pi/2$,



(a) (b) (c) (d)

Figure 6.7: Illustrations for Lemma 6.2. Thick lines drawn inside subtrees represent their legs.

that two of such arms, say S_1 and S_2, are vertical and one, say S_3, horizontal. Consider the possible non-crossing placements of S_1, S_2, and S_3, and consider the lowest horizontal line l intersecting both S_1 and S_2. Two are the cases; either S_3 is below l (Fig. 6.7 (a)), or not (Fig. 6.7 (b)).

In the first case we trivially have $l(\Gamma_h) > l(\Gamma_{h-1}) + l(\Gamma_{h-2})$ (see Fig. 6.7 (c)) and the claim follows.

In the second case we have $l(\Gamma_h) > l(\Gamma_{h-1}) + l(\Gamma_{h-3}) + l(\Gamma_{h-5}) + \ldots + l(\Gamma_{h/2-\lfloor h/2\rfloor+3}) + l(\Gamma_{h/2-\lfloor h/2\rfloor+1})$ (see Fig. 6.7 (d)). However, recurrence

$$(a) : f(x) = f(x-1) + f(x-3) + \ldots + f\left(\frac{x}{2} - \left\lfloor\frac{x}{2}\right\rfloor + 3\right) + f\left(\frac{x}{2} - \left\lfloor\frac{x}{2}\right\rfloor + 1\right)$$

asymptotically provides for $f(x)$ the same value provided by $f(x) = f(x-1) + f(x-2)$. Namely, from (a) we get

$$(b) : f(x-2) = f(x-3) + \ldots + f\left(\frac{x-2}{2} - \left\lfloor\frac{x-2}{2}\right\rfloor + 3\right) + f\left(\frac{x-2}{2} - \left\lfloor\frac{x-2}{2}\right\rfloor + 1\right),$$

that substituted into (a) gives $f(x) = f(x-1) + f(x-2)$.

Hence, $l(\Gamma_h) \geq l(\Gamma_{h-1}) + l(\Gamma_{h-2})$, implying that $l(\Gamma_h)$ grows at least as the terms of the *Fibonacci series*, for which it is well know that the ratio of two consecutive terms l_{k+1} and l_k tends to the *golden ratio* ϕ. Hence $l(\Gamma_h) = \Omega(\phi^h) = \Omega(\phi^{\log_3 n}) = \Omega(n^{1/\log_\phi 3}) = \Omega(n^{0.438})$. The statement follows by observing that the minimum length of the arm of Γ_h grows asymptotically at least as the leg of Γ_h and that each side of Γ_h is at least long as the leg or as the arm of Γ_h. □

In the following we prove that, for complete ternary trees, the lower bound of Lemma 6.2 is tight. Again, we introduce two constructions, called Constructions $\hat{1}$ and $\hat{2}$, defined as follows: Construction $\hat{1}$ has the same geometric inductive step of Construction 1, but the side drawings are recursively constructed with Construction $\hat{2}$ and the base drawing is recursively constructed with Construction $\hat{1}$; Construction $\hat{2}$ has the same geometric inductive step of Construction 2, but the side drawings are recursively constructed with Construction $\hat{1}$ and the base drawing is recursively constructed with Construction $\hat{2}$.

Lemma 6.3 *The drawings built by Construction $\hat{1}$ have $O(n^{0.438})$ height.*

Proof: Denote by H^1_h (by W^2_h) the height (the width) of the drawing of a complete ternary tree with height h built with Construction $\hat{1}$ (with Construction $\hat{2}$).

By simple geometric considerations, we have (1) $H^1_h = \max\{W^2_{h-1}, H^1_{h-1} + (W^2_{h-1}+1)/2\}$ and (2) $W^2_h = \max\{W^2_{h-1}, 2H^1_{h-1}+1\}$.

Suppose by induction that $H^1_{h-1} + (W^2_{h-1}+1)/2 \geq W^2_{h-1}$ and that $2H^1_{h-1} + 1 \geq W^2_{h-1}$. Such inductive hypotheses are verified in the base case, where $H^1_1 = 1$ and $W^2_1 = 1$. Due to the inductive hypotheses, (1) and (2) turn in (1') $H^1_h = H^1_{h-1} + (W^2_{h-1}+1)/2$, and (2') $W^2_h = 2H^1_{h-1}+1$, respectively. We have $H^1_h + (W^2_h+1)/2 = H^1_{h-1} + (W^2_{h-1}+1)/2 + (2H^1_{h-1}+1+1)/2 = 2H^1_{h-1} + (W^2_{h-1})/2 + 3/2 > 2H^1_{h-1} + 1 = W^2_h$, and that $2H^1_h + 1 = 2H^1_{h-1} + W^2_{h-1} + 1 + 1 > 2H^1_{h-1} + 1 = W^2_h$. Hence, the inductive hypothesis is verified and (1') and (2') hold. Substituting (2') into (1'), we get $H^1_h = H^1_{h-1} + ((2H^1_{h-2}+1)+1)/2 = H^1_{h-1} + H^1_{h-2} + 1$. As in the proof of Lemma 6.2, H^1_h grows as the terms of the Fibonacci series, yielding $H^1_h = O(n^{0.438})$. □

6.5 Conclusions and Open Problems

In this chapter we presented the state of the art on the area requirements of straight-line, poly-line, orthogonal, and straight-line orthogonal drawings of trees in small area. We focused our research on straight-line orthogonal drawings of binary and ternary trees. We showed algorithms for constructing order-preserving straight-line orthogonal drawings of binary trees in $O(n^{1.5})$ area, straight-line orthogonal drawings of ternary trees in $O(n^{1.631})$ area, and straight-line orthogonal drawings of complete ternary trees in $O(n^{1.262})$ area. Such upper bounds are, as far as we know, the first sub-quadratic algorithms for the corresponding problems. Further, we have shown lower bounds on the area requirements and on the side lengths of straight-line orthogonal drawings of binary and ternary trees.

As can be noticed from Tables 6.1, 6.2, 6.3, and 6.4 some of the bounds on the area requirements of trees are asymptotically tight, whereas for others there is still space for improvements. In particular, it is a long-standing open question whether all trees admit straight-line/poly-line drawings in linear area.

Open Problem 6.1 *Which are the asymptotic bounds for the area requirements of straight-line and poly-line drawings of trees?*

Concerning straight-line orthogonal drawings, there are still wide gaps between the area upper bounds and the actual lower bounds, especially for order-preserving straight-line orthogonal drawings of binary trees and for straight-line orthogonal drawings of ternary trees. Concerning complete ternary trees, we conjecture that an algorithm only relying on the geometry of Constructions 1 and 2 could improve the upper bound we provided here.

Conjecture 6.1 *Every complete ternary tree admits drawings in $o(n^{1.262})$ area.*

However, we strongly suspect that for (complete) ternary trees linear area straight-line orthogonal drawings are not always achievable. The most fascinating problem in the area of straight-line orthogonal drawings still remains, in our opinion, the one of determining whether binary trees admit straight-line orthogonal drawings in linear area.

Open Problem 6.2 *Which are the asymptotic bounds for the area requirements of straight-line orthogonal drawings of binary trees?*

Chapter 7

Straight-line and Poly-line Upward Drawings of Directed Trees

In this chapter[1] we consider straight-line and poly-line upward drawings of directed trees in small area. We show how to construct optimal $\Theta(n \log n)$ area upward straight-line/poly-line planar drawings of directed trees. However, we prove that if the drawing is required to be order-preserving, then exponential area is required for straight-line upward drawings and quadratic area is required for poly-line upward drawings. Further, we establish tight bounds on the area requirements of planar upward drawings of several families of directed trees, and we show how the results obtained for trees can be exploited to determine asymptotic optimal values for the area requirements of planar upward drawings of directed bipartite graphs and directed outerplanar graphs.

7.1 Introduction

Directed graphs commonly represent hierarchical relationships among objects, such as PERT diagrams, subroutine-call charts, Hasse diagrams, and is-a relationships. When visualizing a directed graph, the drawing is often required

[1] The contents of this chapter appeared in [Fra07a] and in [Fra08b]. Thanks to Walter Didimo and Giuseppe Liotta for reporting the problem of obtaining minimum area upward drawings of directed trees. Thanks to Giuseppe Di Battista for useful discussions.

to be upward. In fact, having all the edges flowing towards the same direction reflects the hierarchical structure of the graph.

Upward drawings of directed acyclic digraphs have been intensively studied from a theoretical point of view. It is known that testing the *upward planarity* of a graph is an *$\mathcal{N}P$-complete* problem if the graph has a variable embedding [GT01], while it is polynomial-time solvable if the embedding of the graph is *fixed* [BDLM94], if the underlying graph is supposed to be an *outerplanar graph* [Pap94], if the digraph has a *single source* [HL96], or if it is a *bipartite* directed acyclic graph [DLR90]. Di Battista and Tamassia showed in [DT88] that a graph is upward planar if and only if it is a subgraph of an *st-planar* graph. Moreover, some families of directed acyclic graphs are always upward planar, like the *series-parallel* digraphs and the digraphs whose underlying graph is a *tree*.

A considerable research effort has been devoted to the design of algorithms for obtaining upward drawings of directed acyclic graphs in small area. Namely, Di Battista, Tamassia, and Tollis have shown in [DTT92] that every upward planar embedded graph can be drawn with upward *poly-line* edges in optimal $\Theta(n^2)$ area, while there exist graphs that require exponential area in any planar *straight-line* upward drawing. Hence, it is natural to restrict the attention to interesting families of directed acyclic graphs, searching for better area bounds. This research direction has been taken by Bertolazzi, Cohen, Di Battista, Tamassia, and Tollis in [BCB$^+$94], where it is shown that *series-parallel* digraphs admit upward planar straight-line drawings in $\Theta(n^2)$ area, while exponential area is generally required if the embedding is chosen in advance.

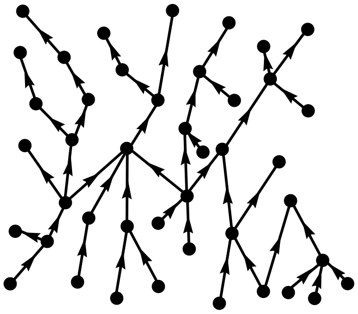


Figure 7.1: A directed tree.

In this chapter we study straight-line and poly-line planar upward drawings of *directed trees*, that is, of directed graphs whose underlying graph is a tree (see Fig. 7.1 for an example of a directed tree), and of some other families of directed acyclic graphs that commonly arise in practice, as directed acyclic graphs whose underlying graph is a bipartite graph (*directed bipartite graphs*), or is an outerplanar graph (*directed outerplanar graphs*). All of such digraph classes exhibit simple and strong structural properties that allow to create planar upward drawings with less constraints and in a easier way with respect to general digraphs. Consequently, we are able to construct straight-line planar upward drawings of directed trees in $\Theta(n \log n)$ area, and to get $\Theta(n)$ area straight-line planar upward drawings for some sub-classes of directed trees. However, we prove that when constraints are imposed on the drawings by forcing an ordering of the neighbors of each vertex, then again exponential area is required for constructing straight-line planar upward drawings and quadratic area is required for constructing poly-line planar upward drawings. Such negative results sharply and surprisingly contrast with the fact that sub-quadratic area is sufficient for constructing straight-line order-preserving upward planar drawings of undirected trees [Cha02, GR03b]. Furthermore, we prove that the lower bounds obtained for directed trees extend also to directed bipartite graphs and directed outerplanar graphs.

More in detail, we provide the following results (throughout the rest of the chapter when we refer to upward drawings we always mean planar upward grid drawings): (i) straight-line and poly-line upward drawings of directed trees can be constructed in optimal $\Theta(n \log n)$ area; (ii) straight-line order-preserving upward drawings of directed trees require (and can be constructed in) exponential area; (iii) poly-line order-preserving upward drawings of directed trees require (and can be constructed in) quadratic area; (iv) *directed binary trees* have the same area requirements of general directed trees; (v) *directed caterpillars* and *directed spider trees* admit linear area straight-line upward drawings; (vi) straight-line upward drawings of directed bipartite graphs require (and can be constructed in) exponential area; (vii) poly-line upward drawings of directed bipartite graphs require (and can be constructed in) quadratic area; (viii) straight-line outerplanar upward drawings of directed outerplanar graphs require (and can be constructed in) exponential area; and (ix) poly-line upward drawings of directed outerplanar graphs require (and can be constructed in) quadratic area.

Tables 7.1 and 7.2 summarize the area requirements of planar upward drawings of directed trees, directed binary trees, directed caterpillars, and directed spider trees.

The rest of the chapter is organized as follows. In Sect. 7.2 we study up-

	Straight-line / Poly-line			
	UB	ref.	LB	ref.
Dir. Trees	$O(n \log n)$	Th. 7.1	$\Omega(n \log n)$	Th. 7.1
Dir. Binary Trees	$O(n \log n)$	Th. 7.1	$\Omega(n \log n)$	Th. 7.1
Dir. Caterpillars	$O(n)$	Th. 7.4	$\Omega(n)$	trivial
Dir. Spider Trees	$O(n)$	Th. 7.5	$\Omega(n)$	trivial

Table 7.1: A table summarizing the results on minimum area non-order-preserving upward drawings of directed trees. Straight-line and poly-line drawings are in the same columns, since they have the same area bounds.

	Straight-line Order-Pres.				Poly-line Order-Pres.			
	UB	ref.	LB	ref	UB	ref.	LB	ref.
Dir. Trees	$O(c^n)$	[GT94]	$\Omega(b^n)$	Th. 7.2	$O(n^2)$	[DTT92]	$\Omega(n^2)$	Th. 7.3
Dir. Binary Trees	$O(c^n)$	[GT94]	$\Omega(b^n)$	Th. 7.2	$O(n^2)$	[DTT92]	$\Omega(n^2)$	Th. 7.3
Dir. Caterpillars	$O(c^n)$	[GT94]	$\Omega(b^n)$	Th. 7.2	$O(n^2)$	[DTT92]	$\Omega(n^2)$	Th. 7.3
Dir. Spider Trees	$O(n)$	Th. 7.5	$\Omega(n)$	trivial	$O(n)$	Th. 7.5	$\Omega(n)$	trivial

Table 7.2: A table summarizing the results on minimum area order-preserving upward drawings of directed trees. b and c denote constants greater than 1.

ward drawings of directed trees; in Sect. 7.3 we study order-preserving upward drawings of directed trees; in Sect. 7.4 we study upward drawings of directed binary trees, directed caterpillars, and directed spider trees; in Sect. 7.5 we study upward drawings of directed bipartite graphs and directed outerplanar graphs; finally, in Sect. 7.6 we conclude and present some open problems.

7.2 Upward Drawings of Trees

In this section we show that directed trees admit straight-line upward drawings in $\Theta(n \log n)$ area and that such an area is necessary in the worst case, even if bends are allowed on the edges.

Concerning the lower bound, Crescenzi *et al.* have shown in [CBP92] a non-directed rooted binary tree T that requires $\Omega(n \log n)$ area in any *strictly upward* grid drawing. Now T can be turned in a directed binary tree T' by directing its edges away from the root. Since an upward drawing of T' is a strictly upward drawing of T, the lower bound on the area requirement of upward drawings of directed trees follows.

Now we show that every directed tree has an $O(n \log n)$ area straight-line upward drawing. This is done by means of an algorithm that considers a directed tree T, removes from T a path called *spine*, recursively draws each disconnected subtree, and finally puts the drawings of the subtrees together with a drawing of the spine, obtaining a drawing of T. Let us describe the algorithm more formally.

Preprocessing: The input is a directed tree T with n nodes. We derive a non-directed rooted tree T' from T by removing the orientations from the edges of T and by choosing a node r in T as root of T'.

Divide: Let T^* be the current non-directed rooted tree and let r^* be its root (at the first step the current tree is T' rooted at r).

If the number of nodes in T^* is greater than one, then select a spine $S^* = (v_0, v_1, \dots, v_k)$ in T^* with the following properties: (i) $v_0 = r^*$, (ii) for $1 \leq i \leq k$, v_i is the root of the heaviest subtree among the subtrees rooted at the children of v_{i-1}, (iii) each edge (v_{i-1}, v_i) is directed from v_i to v_{i-1} in T, for $1 \leq i < k$, and (iv) edge (v_{k-1}, v_k) is directed from v_{k-1} to v_k in T, or v_k is a leaf.

Remove from T^* the nodes of S^*, except for v_k, disconnecting T^* into several non-directed subtrees. We classify such subtrees into sets $T^*(\uparrow, v_i)$ and $T^*(\downarrow, v_i)$, with $0 \leq i < k$, so that a tree rooted at a vertex v goes into set $T^*(\uparrow, v_i)$ (resp. $T^*(\downarrow, v_i)$) if in the directed tree T there is an edge directed from v to v_i (resp. there is an edge directed from v_i to v). Notice that each set could contain several trees. We denote by $T^*(v_k)$ the tree rooted at v_k and by $r(T^*)$ the root of a non-directed tree T^*.

Impera: Assume that in the *Divide* step a tree T^* has been disconnected into a spine S^*, into a subtree $T^*(v_k)$, and into several subtrees in $T^*(\uparrow, v_i)$ and in $T^*(\downarrow, v_i)$, with $0 \leq i < k$. Introduce again the directions on the edges of T^*, obtaining a directed tree $T(v_k)$ from $T^*(v_k)$, obtaining a set $T(\uparrow, v_i)$ of directed trees from the trees in $T^*(\uparrow, v_i)$, and obtaining a set $T(\downarrow, v_i)$ of directed trees from the trees in $T^*(\downarrow, v_i)$. Assume to have for each of such directed trees a drawing satisfying the following properties: $(\mathbf{P}_1)$ the drawing is planar, upward, and straight-line; $(\mathbf{P}_2)$ the root of the tree is placed on the left side of the bounding box of the drawing; and $(\mathbf{P}_3)$ no node of the tree is placed in the drawing below and on the same vertical line of the root of the tree.

Notice that such a drawing can be trivially constructed for a tree with one node. Now we show how to construct a drawing Γ satisfying properties P_1, P_2, and P_3 for the directed tree $\overline{T}$ obtained from T^* by introducing again the directions on the edges. Notice that, in the last *Impera* step, Γ will be a drawing of the whole directed tree T. We distinguish two cases:

k = 1: Place the drawings of the trees in $T(\downarrow, v_0)$ stacked one above the other at one unit of vertical distance, with the left side of their bounding boxes on the same vertical line l, obtaining a drawing Γ'. Place v_0 one unit to the left of l and one unit below $b(\Gamma')$. Place the drawings of the trees in $T(\uparrow, v_0)$ stacked one above the other at one unit of vertical distance, with the left side of their bounding boxes on l, and so that the highest horizontal line intersecting a drawing of a tree in $T(\uparrow, v_0)$ is one unit below v_0, obtaining a drawing Γ''.

If (v_0, v_1) is directed from v_0 to v_1, then place the drawing of $T(v_1)$ so that the left side of its bounding box is on the same vertical line of v_0 and so that the bottom side of its bounding box is one unit above $t(\Gamma'')$ (see Fig. 7.2 (a)). Otherwise, i.e. v_1 is a leaf and (v_0, v_1) is directed from v_1 to v_0, place v_1 on l one unit below $b(\Gamma'')$ (see Fig. 7.2 (b)).

k ≥ 2: Place the drawings of the trees in $T(\downarrow, v_0)$ stacked one above the other at one unit of vertical distance, with the left side of their bounding boxes on the same vertical line l, obtaining a drawing Γ'. Place v_0 two units to the left of l and one unit below $b(\Gamma')$. Place the drawings of the trees in $T(\uparrow, v_0)$ stacked one above the other at one unit of vertical distance, with the left side of their bounding boxes on l, and so that the highest horizontal line intersecting a drawing of a tree in $T(\uparrow, v_0)$ is one unit below v_0, obtaining a drawing Γ_0.

For $i = 1, 2, \ldots, k-2$ (if $k = 2$ this part of the construction is not considered), place the drawings of the trees in $T(\downarrow, v_i)$ stacked one above the other at one unit of vertical distance, with the left side of their bounding boxes on l, and so that the highest horizontal line intersecting a drawing of a tree in $T(\downarrow, v_i)$ is one unit below $b(\Gamma_{i-1})$, obtaining a drawing Γ'_{i-1}. Place v_i one unit to the left of l and and one unit below $b(\Gamma'_{i-1})$. Place the drawings of the trees in $T(\uparrow, v_i)$ stacked one above the other at one unit of vertical distance, with the left side of their bounding boxes on l, and so that the highest horizontal line intersecting a drawing of a tree in $T(\uparrow, v_i)$ is one unit below v_i, obtaining a drawing Γ_i.

Let W be the maximum between the width of the drawing of $T(v_k)$ minus 1 and the maximum width of a drawing of a tree in $T(\uparrow, v_i)$ or in $T(\downarrow, v_i)$ plus 2, with $0 \leq i < k$. Let l' be the vertical line W units to the right of v_0. Mirror the drawings of the trees in $T(\uparrow, v_{k-1})$ with respect to a vertical line and place them stacked one above the other at one unit of vertical distance, with the right side of their mirrored bounding boxes one unit to the left of l' and so that the highest horizontal line intersecting a drawing of a tree in $T(\uparrow, v_{k-1})$ is one unit below $b(\Gamma_{k-2})$. Mirror the drawings of the trees in $T(\downarrow, v_{k-1})$ with respect to a vertical line and place them stacked one above the other at one unit of vertical distance, with the right side of their mirrored bounding boxes

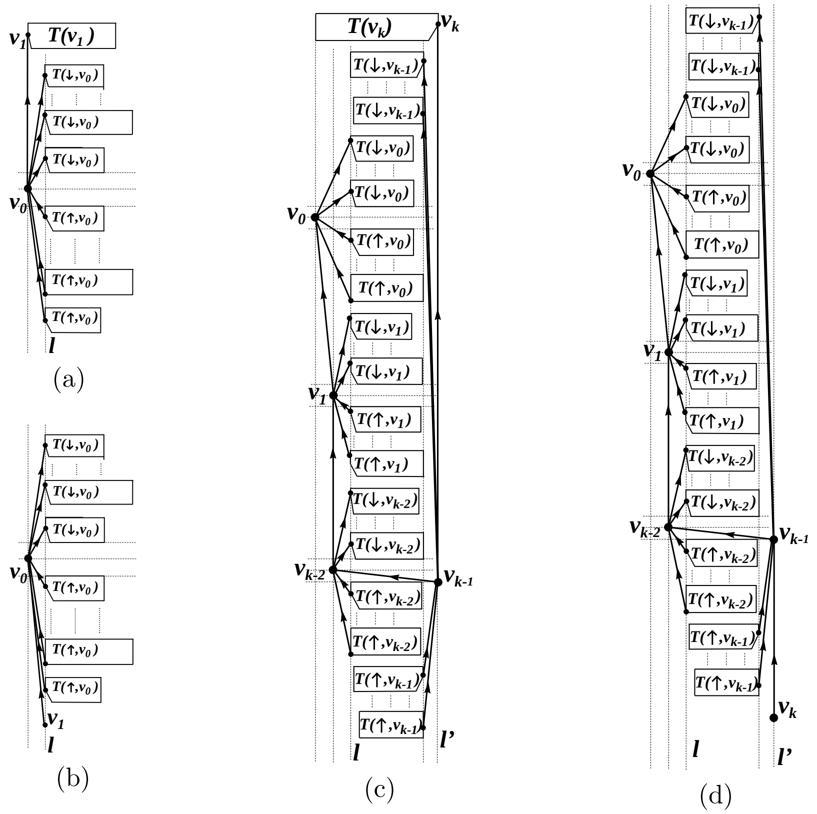


Figure 7.2: *Impera* step of the algorithm for obtaining straight-line non-order preserving upward drawings of trees. (a) $k = 1$ and (v_0, v_1) directed from v_0 to v_1. (b) $k = 1$ and (v_0, v_1) directed from v_1 to v_0. (c) $k > 1$ and (v_{k-1}, v_k) directed from v_{k-1} to v_k. (d) $k > 1$ and (v_{k-1}, v_k) directed from v_k to v_{k-1}.

one unit to the left of l', and so that the lowest horizontal line intersecting a drawing of a tree in $T(\downarrow, v_{k-1})$ is one unit above $t(\Gamma_{k-2})$. Place v_{k-1} on l' one unit below v_{k-2}, obtaining a drawing Γ_{k-1}.

Finally, if edge (v_{k-1}, v_k) is directed from v_{k-1} to v_k, mirror the drawing of $T(v_k)$ with respect to a vertical line and place it with the right side of its mirrored bounding box on l' so that the bottom side of its bounding box is one unit above $t(\Gamma_{k-1})$ (see Fig. 7.2 (c)); otherwise, i.e. v_k is a leaf and edge

(v_{k-1}, v_k) is directed from v_k to v_{k-1}, place v_k on l' one unit below $b(\Gamma_{k-1})$ (see Fig. 7.2 (d)).

The planarity and the upwardness of the final drawing Γ of T can be easily verified. Concerning the area requirements of Γ, the height $h(\Gamma)$ of Γ is $O(n)$, since there is at least one node of the tree for each horizontal line intersecting Γ. Now we compute the width $W(\Gamma)$ of Γ. Denote by $W(T(\uparrow, v_i))$, by $W(T(\downarrow, v_i))$, and by $W(T(v_k))$ the maximum width of the drawing of a tree in $T(\uparrow, v_i)$, of a tree in $T(\downarrow, v_i)$, and of tree $T(v_k)$ constructed by the above described algorithm, respectively. Denote also by $W(n)$ the maximum width of a tree with n nodes constructed by the above described algorithm. Clearly, we have $W(\Gamma) \leq W(n)$.

If $k = 1$ we have $W(\Gamma) = \max\{W(T(v_1)), 1+W(T(\uparrow, v_0)), 1+W(T(\downarrow, v_0))\}$, and if $k \geq 2$ we have $W(\Gamma) = \max_{0 \leq i < k}\{W(T(v_k)), 3+W(T(\uparrow, v_i)), 3+W(T(\downarrow , v_i))\}$. By the definition of S, each tree in $T(\uparrow, v_i)$ and each tree in $T(\downarrow, v_i)$ has at most $n/2$ nodes, and $T(v_k)$ has at most $n - k$ nodes. It follows that $W(n) = \max\{W(n-1), 3 + W(n/2)\}$, that easily solves to $W(n) = O(\log n)$. So we have the following:

Theorem 7.1 *Every n-nodes directed tree admits an upward straight-line drawing in optimal $\Theta(n \log n)$ area.*

7.3 Upward Drawings of Trees with Fixed Embedding

In this section we study the area requirements of order-preserving upward drawings of directed trees. It is easy to observe that a tree T with fixed embedding is upward planar if and only if, for any node $n \in T$, all the outgoing edges of n appear consecutively around n. In such a case the embedding is an *upward planar embedding*. In the following we refer only to upward planar embeddings. Garg and Tamassia proved in [GT94] that any upward planar embedding can be realized with straight-line edges in exponential area. Hence, exponential area straight-line upward drawings of embedded directed trees are feasible.

Now we prove the claimed exponential lower bound. Bertolazzi *et al.* showed in [BCB$^+$94] an embedding $\mathcal{E}_n$ of a $2n$-vertex *series-parallel* digraph requiring $\Omega(4^n)$ area in any order-preserving upward straight-line drawing. Such an embedding is recursively defined as follows: $\mathcal{E}_0$ consists of a single edge (s_0, t_0); $\mathcal{E}_{n+1}$ is obtained from $\mathcal{E}_n$ by adding (i) two new nodes s_{n+1} and t_{n+1}, (ii) an edge from s_{n+1} to s_n, (iii) an edge from t_n to t_{n+1}, (iv) an edge from s_n to t_{n+1} to the right of $\mathcal{E}_n$, and (v) an edge from s_{n+1} to t_{n+1} to the left of $\mathcal{E}_n$ (see Fig. 7.3 (a)).

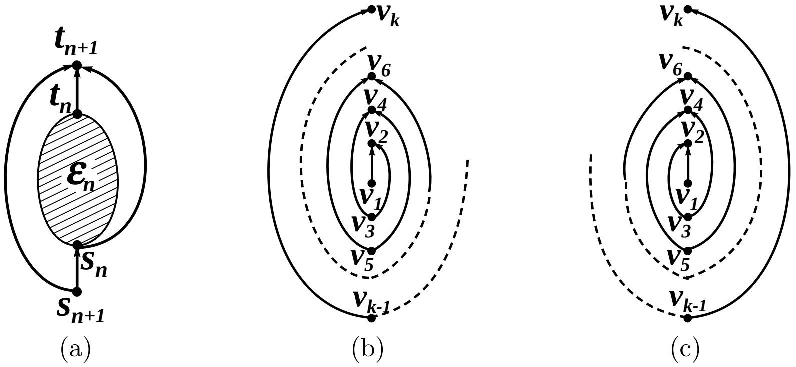


Figure 7.3: (a) Embedding $\mathcal{E}_{n+1}$ of a series-parallel digraph requiring exponential area. (b) A clockwise coil. (c) A counter-clockwise coil.

We define a *clockwise coil* S to be an upward planar drawing of a directed path $P = (v_1, v_2, \ldots, v_k)$ satisfying the following properties: **p**roperty (i) the edges (v_i, v_{i+1}) of P, with i odd (with i even), are directed from v_i to v_{i+1} (resp. from v_{i+1} to v_i), **p**roperty (ii) $y(v_i) < y(v_j)$ ($y(v_i) > y(v_j)$), for every i odd (resp. for every i even) and every j such that $j < i$, and **p**roperty (iii) for every i odd (for every i even) all the vertices v_j such that $j < i$ are contained in the open region $R(v_i, v_{i+1})$ delimited by the edge (v_i, v_{i+1}) and by the horizontal half-lines starting at v_i and at v_{i+1} and directed toward increasing x-coordinates (resp. toward decreasing x-coordinates) (see Fig. 7.3 (b)). A *counter-clockwise coil* is defined analogously, with *odd* replaced by *even* and vice-versa in property (iii) (see Fig. 7.3 (c)). We have:

Lemma 7.1 *Any straight-line n-vertex clockwise or counter-clockwise coil requires $\Omega(2^n)$ area.*

Proof: Consider any straight-line clockwise coil S. We show that adding segments (v_i, v_{i+2}), for $i = 1, 2 \ldots, n-2$, augments S in a planar drawing S'. Namely, we prove that a segment (v_i, v_{i+2}) does not intersect (a) any segment (v_j, v_{j+1}) of S, with $j \leq i$, (b) segment (v_{i+1}, v_{i+2}) of S, (c) segment (v_{i+2}, v_{i+3}) of S, (d) any segment (v_j, v_{j+1}) of S, with $j > i+2$, and (e) any segment (v_j, v_{j+2}), with $j \neq i$ added to S.

(a) Suppose i is odd (is even). By property (ii) no vertex v_j of S, with $j < i+2$ and $j \neq i$, lies in the open half-plane $\mathcal{H}$ below (resp. above) the horizontal line through v_i. Moreover, v_{i+2} is contained in $\mathcal{H}$. Hence, (v_i, v_{i+2}) does not create crossings with any segment (v_j, v_{j+1}) of S, with $j \leq i$.

(b) Since they are adjacent, (v_i, v_{i+2}) and (v_{i+1}, v_{i+2}) cross only if they overlap. But in such a case (v_i, v_{i+1}) and (v_{i+1}, v_{i+2}) overlap, too. However, this is not possible by the supposed planarity of S.

(c) Since they are adjacent, (v_i, v_{i+2}) and (v_{i+2}, v_{i+3}) cross only if they overlap. However, by property (iii) v_i is contained inside $R(v_{i+2}, v_{i+3})$. Hence (v_i, v_{i+2}) is internal to $R(v_{i+2}, v_{i+3})$, except for vertex v_{i+2}, and cannot overlap with (v_{i+2}, v_{i+3}) that is, by definition, on the border of $R(v_{i+2}, v_{i+3})$.

(d) By property (iii) v_i and v_{i+2} are contained inside $R(v_j, v_{j+1})$, so (v_i, v_{i+2}) is internal to $R(v_j, v_{j+1})$ and cannot cross (v_j, v_{j+1}) that is, by definition, on the border of $R(v_j, v_{j+1})$.

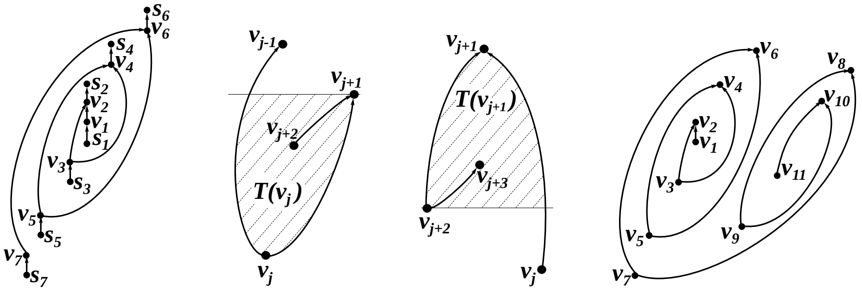
(e) It is easy to see that segments (v_i, v_{i+2}), for $i = 1, 2 \ldots, n-2$, form a directed path with increasing y-coordinate and so they don't cross each other.

Now one can observe that S' is an upward drawing of $\mathcal{E}_{n/2}$ (see [BCB$^+$94] and the beginning of the section). Hence, an n-vertex straight-line clockwise coil S requires the same area of a straight-line drawing of $\mathcal{E}_{n/2}$, that is $\Omega(4^{n/2}) = \Omega(2^n)$. If S is a counter-clockwise straight-line coil a straightforward modification of the previous proof shows that S requires $\Omega(2^n)$ area. □

Now let T^* be a tree composed of an $n/2$-nodes path $P^* = (v_1, v_2, \ldots, v_{n/2})$ and of $n/2$ leaves s_i, $1 \leq i \leq n/2$, such that s_i adjacent to v_i. Suppose, for simplicity of notation, that n and $n/2$ are even. Edges (v_i, v_{i+1}), with i odd (with i even), are directed from v_i to v_{i+1} (resp. from v_{i+1} to v_i). Edges (v_i, s_i), with i odd (with i even), are directed from s_i to v_i (resp. from v_i to s_i). We fix for T^* an embedding $\mathcal{E}^*$ such that, for each node v_i with $2 \leq i \leq n/2$, the clockwise order of the edges incident in v_i is $[s_i, v_{i-1}, v_{i+1}]$ (see Fig. 7.4 (a)). We claim the following:

Lemma 7.2 *Every upward drawing Γ^* of T^* with embedding $\mathcal{E}^*$ contains a clockwise or a counter-clockwise coil of at least $n/4$ nodes.*

Proof: Observe that, by the embedding constraints of $\mathcal{E}^*$ and by the upwardness of Γ^*, path P^* turns in counter-clockwise direction at every edge (v_{i-1}, v_i), for $i = 2, 3, \ldots, n/2$, i. e., considering the half-lines t_1 and t_2 starting at v_i and tangent to the curves representing edges (v_{i-1}, v_i) and (v_i, v_{i+1}), respectively, the angle described by a counter-clockwise movement that leads t_2 to overlap with t_1 is less than π. Let j be the highest index such that the drawing S^*_1 of the subpath $(v_1, v_2, \ldots, v_j)$ of P^* is a counter-clockwise coil.



(a) (b) (c) (d)

Figure 7.4: (a) An upward drawing of T^* with embedding $\mathcal{E}^*$. (b) $T(v_j)$. (c) $T(v_{j+1})$. (d) An upward drawing of P^*. Notice that $(v_1, v_2, \ldots, v_7)$ is a counter-clockwise coil, while $(v_{11}, v_{10}, \ldots, v_7)$ is a clockwise coil.

If $j \geq n/4$ the lemma follows. Otherwise, we claim that the drawing S_2^* of the subpath $(v_{n/2}, v_{n/2-1}, \ldots, v_{j+1}, v_j)$ of P^* is a clockwise coil.

Assume j is odd. Property (i) follows from the definition of T^* and from the upwardness of Γ^*. To prove that S_2^* satisfies property (ii), consider three vertices v_{i-1}, v_i, and v_{i+1} that are consecutive in S_2^*. Let v_t be the one between v_{i-1} and v_{i+1} such that $|y(v_i) - y(v_t)|$ is minimum. Denote by $T(v_i)$, with $i = j, j+1, \ldots, n/2-1$ the triangle with curved edges delimited by (v_i, v_{i-1}), by (v_i, v_{i+1}), and by the horizontal line through v_t. Since $(v_1, v_2, \ldots, v_j, v_{j+1})$ is not a coil, then $y(v_{j-1}) \geq y(v_{j+1})$. Since (v_{j+1}, v_{j+2}) turns in clockwise direction with respect to (v_j, v_{j+1}), the planarity and the upwardness of Γ^* imply that v_{j+2} is inside $T(v_j)$, and so $y(v_{j+2}) > y(v_j)$ (see Fig. 7.4 (b)). Since (v_{j+2}, v_{j+3}) turns in clockwise direction with respect to (v_{j+1}, v_{j+2}), the planarity and the upwardness of Γ^* imply that v_{j+3} is inside $T(v_{j+1})$, and so $y(v_{j+3}) < y(v_{j+1})$ (see Fig. 7.4 (c)). Proceeding in the same way, it follows that, for all $i = j, j+1, \ldots, n/2-2$, $y(v_{i+2}) > y(v_i)$ (resp. $y(v_{i+2}) < y(v_i)$) with i odd (resp. with i even). Hence, property (ii) is satisfied by S_2^*. Further, property (iii) is satisfied by S_2^*, since every vertex v_k, with $k \geq i+2$ is contained inside $T(v_i)$ and, consequently, inside $R(v_i, v_{i+1})$, that encloses $T(v_i)$. If j is even an analogous proof shows that S_2^* is a clockwise coil. Finally, since $j < n/4$, S_2^* contains at least $n/2 - j > n/4$ nodes. □

Theorem 7.2 *There exists an n-nodes embedded directed tree requiring $\Omega(b^n)$ area, with b greater than 1, in any upward straight-line order-preserving draw-*

ing.

Proof: Consider T^* and its embedding $\mathcal{E}^*$ described in this section. By Lemma 7.2 every upward drawing of T^* with embedding $\mathcal{E}^*$ contains a coil of at least $n/4$ nodes that, by Lemma 7.1, requires $\Omega(2^{n/4}) = \Omega((\sqrt[4]{2})^n) = \Omega(b^n)$, with $b = \sqrt[4]{2}$. □

Now we turn to poly-line drawings. Di Battista *et al.* have shown in [DTT92] that every upward planar embedding can be drawn with poly-line edges in $O(n^2)$ area. It follows that quadratic area poly-line upward drawings of embedded directed trees are feasible. Concerning the lower bound, we have the following:

Lemma 7.3 *Any n-vertex poly-line clockwise or counter-clockwise coil requires $\Omega(n^2)$ area.*

Proof: By property (ii) vertex v_i, with i odd, has y-coordinate less than the one of every vertex v_j, with $j < i$. This implies that $n/2$ vertices v_i such that i is odd occupy $n/2$ distinct horizontal lines and so the height of S is $\Omega(n)$. Concerning the width of S, suppose w.l.o.g. to draw S starting from a drawing Γ_1 of v_1, and then iteratively constructing a drawing Γ_i by adding vertex v_i and edge (v_{i-1}, v_i) to Γ_{i-1}, for $i = 2, \ldots, n$. We claim that the width of Γ_i is at least the width of Γ_{i-2} plus one. Suppose that the width of Γ_i is at most the width of Γ_{i-2}. Then Γ_{i-2} cannot be enclosed inside $R(v_{i-1}, v_i)$ and so property (iii) cannot be satisfied. It follows that the width of S is $\Omega(n)$. □

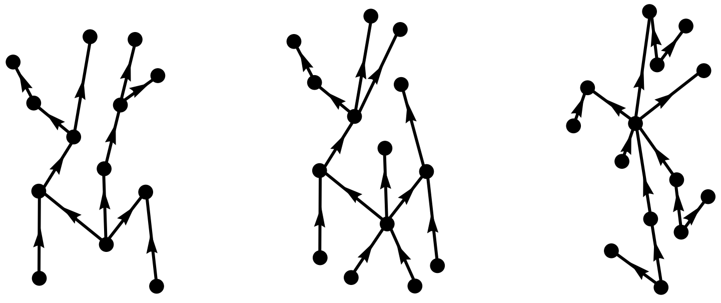
Hence, we can again consider directed tree T^* with fixed embedding $\mathcal{E}^*$. By Lemma 7.2 every upward drawing of T^* with embedding $\mathcal{E}^*$ contains a clockwise or a counter-clockwise coil S of at least $n/4$ nodes. By Lemma 7.3 quadratic area is required for S.

Theorem 7.3 *There exists an n-nodes directed tree T^* and an embedding of T^* requiring $\Omega(n^2)$ area in any upward poly-line order-preserving drawing.*

7.4 Upward Drawings of Some Families of Directed Trees

In this section we study the area requirements of planar upward drawings of some families of directed trees, like *directed binary trees* (see Fig. 7.5 (a)), *directed caterpillars* (see Fig. 7.5 (b)), and *directed spider trees* (see Fig. 7.5

(c)), searching for better area bounds with respect to those obtained for general trees.



(a) (b) (c)

Figure 7.5: (a) A directed binary tree. (b) A directed caterpillar. (c) A directed spider tree.

Concerning *directed binary trees*, one can observe that the lower bounds on the area requirement of planar upward drawings of directed trees presented in Sections 7.2 and 7.3 are obtained by considering directed *binary* trees. Hence such lower bounds are still valid here. Moreover, the algorithms for drawing directed trees clearly apply also to directed binary trees, hence the optimal bounds on the area requirement of planar upward drawings of directed binary trees are the same of the ones of general trees.

Analogously, concerning *directed caterpillars*, we notice that the lower bounds on the area requirement of order-preserving upward drawings of directed trees presented in Section 7.3 were obtained by considering a directed caterpillar. Hence such lower bounds are still valid here. On the other hand, for non-order-preserving drawings one can obtain better results with respect to those for general trees, as shown by the following:

Theorem 7.4 *Every n-nodes directed caterpillar admits an upward straight-line drawing in optimal $\Theta(n)$ area.*

Proof: Let C be an n-nodes directed caterpillar and let $P = (v_1, v_2, \ldots, v_m)$ be the path obtained from C by removing its leaves. Let v_1 be the root of C. We will denote by $C(v_i)$ the caterpillar obtained from C by removing nodes $v_1, \ldots, v_{i-1}$ and their adjacent leaves. We prove by induction on m that an upward drawing Γ of C satisfying the following properties can be constructed:

($\mathbf{P}_1$) the root of C is placed on $l(\Gamma)$; ($\mathbf{P}_2$) no node of C is placed below and on the same vertical line of the root of C; ($\mathbf{P}_3$) the width of Γ is less or equal than 4; ($\mathbf{P}_4$) the height of Γ is less or equal than n.

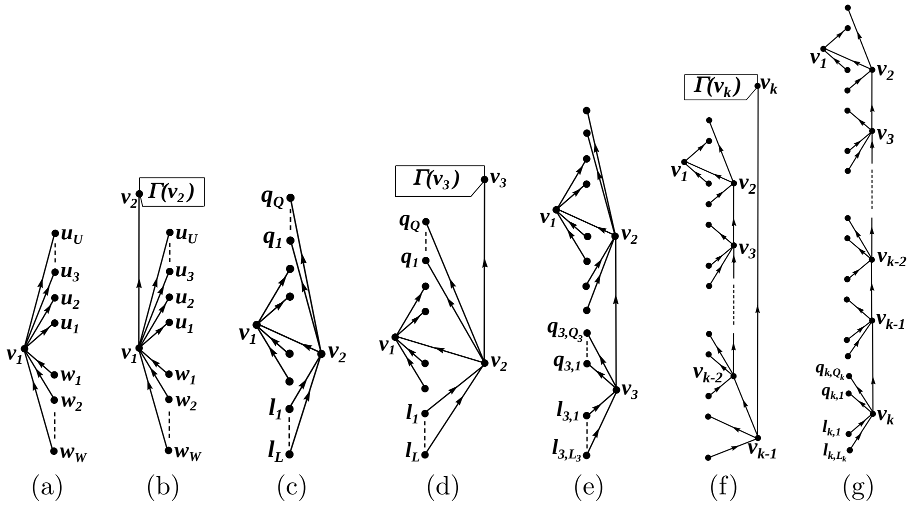


Figure 7.6: Illustration of the algorithm for constructing planar upward drawings of caterpillars. (a) $m = 1$. (b) $k = 2$ and edge (v_1, v_2) directed from v_1 to v_2. (c) $k = m = 2$ and edge (v_1, v_2) directed from v_2 to v_1. (d) $k = 3$ and edge (v_2, v_3) directed from v_2 to v_3. (e) $k = m = 3$ and edge (v_2, v_3) directed from v_3 to v_2. (f) $k > 3$ and edge (v_{k-1}, v_k) directed from v_{k-1} to v_k. (g) $k = m$ and edge (v_{k-1}, v_k) is directed from v_k to v_{k-1}.

Let $u_1, u_2, \dots, u_U$ be the neighbors of v_1 such that C contains edges directed from v_1 to u_i. Further, let $w_1, w_2, \dots, w_W$ be the neighbors of v_1 such that C contains edges directed from w_i to v_1. Place v_1 in $(0, 0)$, place $u_1, u_2, \dots, u_U$ on the line $x = 1$ with distinct integer y-coordinates between 1 and U, and place $w_1, w_2, \dots, w_W$ on the line $x = 1$ with distinct integer y-coordinates between -1 and $-W$. If $m = 1$ the algorithm stops (see Fig. 7.6 (a)).

If $m > 1$, select vertex v_k in P such that: (i) each edge (v_{i-1}, v_i) of P is directed from v_i to v_{i-1}, for $1 < i < k$, and (ii) edge (v_{k-1}, v_k) is directed from v_{k-1} to v_k or $k = m$.

If $k = 2$ and edge (v_1, v_2) is directed from v_1 to v_2, recursively construct a drawing $\Gamma(v_2)$ of $C(v_2)$ and place $\Gamma(v_2)$ so that $l(\Gamma(v_2))$ lies on the axis $x = 0$

and so that $b(\Gamma(v_2))$ lies on the horizontal line $y = U + 1$ (see Fig. 7.6 (b)).

If $k = m = 2$ and edge (v_1, v_2) is directed from v_2 to v_1, then let $q_1, q_2, \ldots, q_Q$ be the neighbors of v_2 such that C contains edges directed from v_2 to q_i. Further, let $l_1, l_2, \ldots, l_L$ be the neighbors of v_2 such that C contains edges directed from l_i to v_2. Place $q_1, q_2, \ldots, q_Q$ on the line $x = 1$ with distinct integer y-coordinates between $U + 1$ and $U + Q$, and place $l_1, l_2, \ldots, l_L$ on the line $x = 1$ with distinct integer y-coordinates between $-W - 1$ and $-W - L$. Place v_2 at grid point $(2, -1)$ (see Fig. 7.6 (c)).

If $k > 2$, let $q_1, q_2, \ldots, q_Q$ be the neighbors of v_2 such that C contains edges directed from v_2 to q_i. Further, let $l_1, l_2, \ldots, l_L$ be the neighbors of v_2 such that C contains edges directed from l_i to v_2. Place $q_1, q_2, \ldots, q_Q$ on the line $x = 1$ with distinct integer y-coordinates between $U + 1$ and $U + Q$, and place $l_1, l_2, \ldots, l_L$ on the line $x = 1$ with distinct integer y-coordinates between $-W - 1$ and $-W - L$. If $k = 3$ and edge (v_2, v_3) is directed from v_2 to v_3, place v_2 at grid point $(3, -1)$, recursively construct a drawing $\Gamma(v_3)$ of $C(v_3)$ and mirror $\Gamma(v_3)$ with respect to a vertical axis, so that v_3 lies on $r(\Gamma(v_3))$. Place the mirrored $\Gamma(v_3)$ so that $r(\Gamma(v_3))$ lies on the vertical line $x = 3$ and so that $b(\Gamma(v_3))$ lies on the horizontal line $y = U + Q + 1$ (see Fig. 7.6 (d)).

If $k = m = 3$ and edge (v_2, v_3) is directed from v_3 to v_2, let $q_{3,1}, q_{3,2}, \ldots, q_{3,Q_3}$ be the neighbors of v_3 such that C contains edges directed from v_3 to $q_{3,j}$. Further, let $l_{3,1}, l_{3,2}, \ldots, l_{3,L_3}$ be the neighbors of v_3 such that C contains edges directed from $l_{3,j}$ to v_3. Place v_2 at grid point $(2, -1)$, place $q_{3,1}, q_{3,2}, \ldots, q_{3,Q_3}$ on the line $x = 1$ with distinct integer y-coordinates between $-W - L - 1$ and $-W - L - Q_3$, place v_3 at grid point $(2, -W - L - Q_3 - 1)$, and place $l_{3,1}, l_{3,2}, \ldots, l_{3,L_3}$ on the line $x = 1$ with distinct integer y-coordinates between $-W - L - Q_3 - 2$ and $-W - L - Q_3 - L_3 - 1$ (see Fig. 7.6 (e)).

If $k > 3$ place v_2 at grid point $(2, -1)$. The drawing of C will be constructed by iteratively adding to the drawing constructed so far vertex v_i and its adjacent leaves, for $i = 3, \ldots, k - 1$. Denote by y_i the minimum y-coordinate that has been already assigned to a node when vertex v_i and its adjacent leaves have to be added to the drawing. Let $q_{i,1}, q_{i,2}, \ldots, q_{i,Q_i}$ be the neighbors of v_i such that C contains edges directed from v_i to $q_{i,j}$. Further, let $l_{i,1}, l_{i,2}, \ldots, l_{i,L_i}$ be the neighbors of v_i such that C contains edges directed from $l_{i,j}$ to v_i. Place $q_{i,1}, q_{i,2}, \ldots, q_{i,Q_i}$ on the line $x = 1$ with distinct integer y-coordinates between $y_i - 1$ and $y_i - Q_i$, place v_i at grid point $(2, y_i - Q_i - 1)$, and place $l_{i,1}, l_{i,2}, \ldots, l_{i,L_i}$ on the line $x = 1$ with distinct integer y-coordinates between $y_i - Q_i - 2$ and $y_i - Q_i - L_i - 1$. If edge (v_{k-1}, v_k) is directed from v_{k-1} to v_k, shift vertex v_{k-1} one unit to the right, recursively construct a drawing $\Gamma(v_k)$ of $C(v_k)$ and mirror $\Gamma(v_k)$ with respect to a vertical axis, so that v_k lies on

$r(\Gamma(v_k))$. Place the mirrored $\Gamma(v_k)$ so that $r(\Gamma(v_k))$ lies on the vertical line $x = 3$ and so that $b(\Gamma(v_k))$ lies on the horizontal line $y = U + Q + 1$ (see Fig. 7.6 (f)).

If $k = m$ and edge (v_{k-1}, v_k) is directed from v_k to v_{k-1}, let $q_{k,1}, q_{k,2}, \ldots, q_{k,Q_k}$ be the neighbors of v_k such that C contains edges directed from v_k to $q_{k,j}$. Further, let $l_{k,1}, l_{k,2}, \ldots, l_{k,L_k}$ be the neighbors of v_k such that C contains edges directed from $l_{k,j}$ to v_k. Place $q_{k,1}, q_{k,2}, \ldots, q_{k,Q_k}$ on the line $x = 1$ with distinct integer y-coordinates between $y_k - 1$ and $y_k - Q_k$, place v_k at grid point $(2, y_k - Q_k - 1)$, and place $l_{k,1}, l_{k,2}, \ldots, l_{k,L_k}$ on the line $x = 1$ with distinct integer y-coordinates between $y_k - Q_k - 2$ and $y_k - Q_k - L_k - 1$ (see Fig. 7.6 (g)).

It is easy to verify that the constructed drawing satisfies properties $(\mathbf{P}_1)$ and $(\mathbf{P}_2)$. Moreover, since all the vertices are assigned x-coordinates between 0 and 3, and since there is at least one vertex for every horizontal line intersecting the drawing, then also properties $(\mathbf{P}_3)$ and $(\mathbf{P}_4)$ are satisfied. □

For *directed spider trees* linear area is achievable even for order-preserving drawings:

Theorem 7.5 *Every n-nodes directed spider tree admits an upward order-preserving straight-line drawing in optimal $\Theta(n)$ area.*

Proof: Denoting by k the degree of v, S is composed of v and of k paths all incident to v. We now show how to construct a linear-area straight-line order-preserving planar upward drawing of S. Suppose that an embedding $\mathcal{E}$ is specified for S by means of a clockwise order $v_1, v_2, \ldots, v_k$ of the neighbors of v. Since a planar upward drawing of S exists if and only if all the outgoing edges of v appear consecutively around v, we can suppose w.l.o.g. that there exists an index q, with $1 \leq q \leq k$, such that all the edges (v, v_i), with $1 \leq i \leq q$, are directed from v to v_i and all the edges (v, v_i), with $q + 1 \leq i \leq k$, are directed from v_i to v. Let P_i be the path starting at v and containing vertex v_i.

Di Giacomo, Liotta, Meijer, and Wismath proved in [DLMW05] that every directed path admits a straight-line upward drawing in a bounding box that has height $O(n)$ and width 3. Moreover, the drawings constructed by the algorithm shown by Di Giacomo *et al.* [DLMW05] have one end-point of the path placed on the left side of the bounding box of the drawing. Hence, a straight-line upward drawing of S can be obtained as follows: Construct a drawing Γ_i of each path $P_i \setminus v$, $1 \leq i \leq k$, with v_i on $l(\Gamma_i)$, as described in [DLMW05]; place the drawings Γ_i stacked one above the other, so that $b(\Gamma_i)$ is one unit above $t(\Gamma_{i+1})$, with $1 \leq i \leq q - 1$ and $q + 1 \leq i \leq k - 1$, so that $b(\Gamma_q)$ is two units

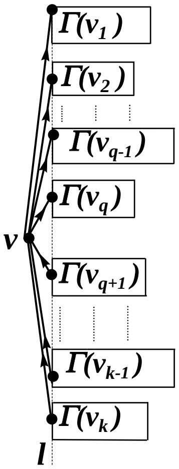


Figure 7.7: Illustration of the algorithm for constructing planar upward drawings of spider trees.

above $t(\Gamma_{q+1})$, and so that all the $l(\Gamma_i)$ lie on the same vertical line l; finally insert v one unit to the left of l and one unit below $b(\Gamma_q)$ and draw edges between v and its neighbors as straight-line segments (see Fig. 7.7).

It is easy to see that the constructed drawing Γ is order-preserving, planar, and upward. Moreover, the width of Γ is 4, and its height is one plus the sum of the heights of the Γ_i's, hence it is linear. □

7.5 Upward Drawings of Directed Bipartite and Outerplanar Graphs

In this section we consider upward drawings of families of directed acyclic graphs richer than directed trees, and we show that exponential area for straight-line drawings and quadratic area for poly-line drawings are sometimes necessary even without forcing an order of the neighbors of each vertex.

In the following we show the inductive construction of an n-vertex directed bipartite graph B_n. Such a digraph contains an $\Omega(n)$ nodes coil in any upward planar drawing, hence it requires exponential area in any straight-line upward drawing and quadratic area in any poly-line upward drawing. Such lower bounds are again matched by the upper bounds in [GT94] and in [BCB$^+$94].

We define B_n as the directed bipartite graph with vertex sets V and U, inductively defined as follows: (i) B_8 has vertices $v_{-2}, v_{-1}, v_1, v_2 \in V$ and $u_{-2}, u_{-1}, u_1, u_2 \in U$, the edges of a directed path $(v_{-2}, u_{-2}, v_{-1}, u_{-1}, v_1, u_1, v_2, u_2)$, and the directed edges (v_1, u_2), (v_{-1}, u_1), (v_{-2}, u_1) and (v_{-1}, u_2) (see Fig. 7.8 (a)); (ii) B_n, with n multiple of 4, is constructed from B_{n-4}, by adding four new vertices $v_{n/4}$, $u_{n/4}$, $v_{-n/4}$, and $u_{-n/4}$ and eight directed edges $(v_{-n/4}, u_{-n/4})$, $(u_{-n/4}, v_{-n/4+1})$, $(u_{n/4-1}, v_{n/4})$, $(v_{n/4}, u_{n/4})$, $(v_{-n/4+2}, u_{n/4})$, $(v_{-n/4+1}, u_{n/4-1})$, $(v_{-n/4}, u_{n/4-1})$, and $(v_{-n/4+1}, u_{n/4})$ (see Fig. 7.8 (b)). We claim the following:

Theorem 7.6 *There exists an n-vertex directed bipartite graph requiring $\Omega(b^n)$ area, with b greater than 1, in any upward straight-line drawing.*

Theorem 7.7 *There exists an n-vertex directed bipartite graph requiring $\Omega(n^2)$ area in any upward poly-line drawing.*

In the following we prove the previous theorems, by considering the n-vertex directed bipartite graph B_n. The outline of the proof is as follows: First, we prove that B_n has a unique planar upward embedding $\mathcal{E}_n$, up to a reversal of the adjacency list of each vertex; second, we show that every planar upward drawing of B_n with embedding $\mathcal{E}_n$ contains an $\Omega(n)$ nodes clockwise or counter-clockwise coil; by Lemmas 7.1 and 7.3 this is sufficient to prove Theorems 7.6 and 7.7.

Lemma 7.4 *B_n has exactly one upward planar embedding $\mathcal{E}_n$, up to a reversal of the adjacency list of each vertex.*

Proof: Assume w.l.o.g. to construct an upward drawing of B_n starting from a drawing $\mathcal{P}$ of the directed path

$$(v_{-n/4}, u_{-n/4}, v_{-n/4+1}, u_{-n/4+1}, \dots, v_{-1}, u_{-1}, v_1, u_1, v_2, u_2, \dots, v_{n/4}, u_{n/4})$$

and by iteratively adding the remaining edges. (v_1, u_2) can be drawn to the left or to the right of $\mathcal{P}$. Suppose that it is drawn to the left. Then (v_{-1}, u_1) and (v_{-2}, u_1) must be drawn to the right of $\mathcal{P}$, since u_1 is closed on the left by (v_1, u_2). Analogously, (v_{-1}, u_2) must be drawn to the left of $\mathcal{P}$, since v_{-1} is closed on the right by (v_{-2}, u_1). Inductively, we can assume that if (v_1, u_2) is drawn to the left of $\mathcal{P}$ there is only one embedding $\mathcal{E}_{n-4}$ for B_{n-4}. Let now draw the remaining edges of B_n. Edge $(v_{-n/4+2}, u_{n/4})$ must be drawn to the left of $\mathcal{P}$, since $v_{-n/4+2}$ is closed on the right from $(v_{-n/4+1}, u_{n/4-2})$.

Then $(v_{-n/4+1}, u_{n/4-1})$ and $(v_{-n/4}, u_{n/4-1})$ must be drawn to the right of $\mathcal{P}$, since $u_{n/4-1}$ is closed on the left from $(v_{-n/4+2}, u_{n/4})$. Finally $(v_{-n/4+1}, u_{n/4})$ must be drawn to the left of $\mathcal{P}$, since $v_{-n/4+1}$ is closed on the right from edge $(v_{-n/4+2}, u_{n/4})$. Hence, embedding (v_1, u_2) to the left of $\mathcal{P}$ fully determines the embedding $\mathcal{E}_n$ of B_n. An analogous proof shows that embedding (v_1, u_2) to the right of $\mathcal{P}$ fixes the same embedding $\mathcal{E}_n$ of B_n, but with the adjacency list of each vertex reverted with respect to the previous case. □

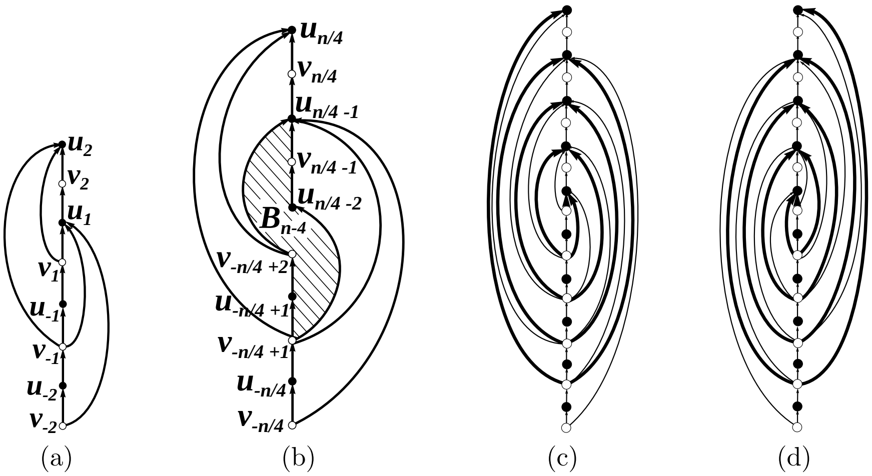


Figure 7.8: (a) B_8. (b) B_n, defined in terms of B_{n-4}. (c) and (d) The unique embeddings of B_{20}, when (v_1, u_2) is drawn to the left or to the right of $\mathcal{P}$, respectively. The thick lines show the coil S.

Lemma 7.5 *Any upward drawing* Γ_n *of* $\mathcal{E}_n$ *contains an* $n/2$*-vertex clockwise or counter-clockwise coil.*

Proof: Consider the drawing S of the path

$$(v_1, u_1, v_{-1}, u_2, v_{-2}, u_3, \ldots, v_{-i+1}, u_i, \ldots, v_{-n/4+1}, u_{n/4-1}, v_{-n/4})$$

in Γ_n. We claim that S is a clockwise or a counter-clockwise coil. Property (i) is satisfied for S because of the definition of $\mathcal{E}_n$ and of the upwardness of Γ_n. It can be easily checked that property (ii) is satisfied because $\mathcal{P}$ fully determines the y-ordering of the vertices of B_n. It can be proved that property

(iii) is satisfied for S by the same considerations that constitute the proof of Lemma 7.4, where if edge (v_1, u_2) is drawn to the left (to the right) of $\mathcal{P}$ then S is a clockwise (resp. counter-clockwise) coil. Figures 7.8 (c) and 7.8 (d) show the coil S in the embedding $\mathcal{E}_{20}$ of B_{20}. □

Hence, by Lemmas 7.4 and 7.5 digraph B_n has only one upward planar embedding that contains an $n/2$-vertex coil. By Lemma 7.1 every straight-line n-vertex coil requires $\Omega(2^n)$ area. This implies that every straight-line drawing of B_n requires $\Omega(2^{n/2}) = \Omega((\sqrt{2})^n) = \Omega(b^n)$ area, with $b = \sqrt{2}$, proving Theorem 7.6. Further, by Lemma 7.3 every poly-line n-vertex coil, and so B_n, requires $\Omega(n^2)$ area, proving Theorem 7.7.

Again using arguments based on the results obtained for directed trees, it can be shown that *directed outerplanar graphs* generally require exponential area in any outerplanar straight-line upward drawing and quadratic area in any poly-line upward drawing.

Consider the n-vertex directed outerplanar graph O_n inductively defined as follows: (i) O_4 has four vertices v_1, v_2, v_3, and v_4 and four directed edges (v_1, v_2), (v_1, v_4), (v_2, v_3), and (v_3, v_4) (see Fig. 7.9 (a)); (ii) O_{n+4} is composed of O_n, of four new vertices v_{n+1}, v_{n+2}, v_{n+3}, and v_{n+4}, and of six new directed edges (v_{n+1}, v_n), (v_{n+2}, v_{n-1}), (v_{n+1}, v_{n+2}), (v_{n+2}, v_{n+3}), (v_{n+1}, v_{n+4}), and (v_{n+3}, v_{n+4}) (see Fig. 7.9 (b)).

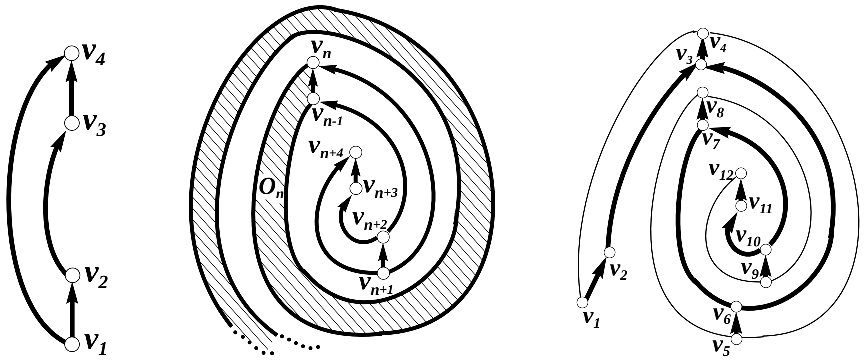
We claim the following theorems:

Theorem 7.8 *There exists an n-vertex directed outerplanar graph requiring $\Omega(b^n)$ area, with b greater than 1, in any upward outerplanar straight-line drawing.*

Theorem 7.9 *There exists an n-vertex directed outerplanar graph requiring $\Omega(n^2)$ area in any upward poly-line drawing.*

To prove Theorem 7.8, consider the directed outerplanar graph O_n. Since O_n is a biconnected outerplanar graph, then it has only one outerplanar embedding $\mathcal{O}_n$, up to a reversal of the adjacency list of each vertex. It is easy to see that $\mathcal{O}_n$ contains an embedding $\mathcal{E}^*$ of tree T^* as subgraph (see Fig. 7.9 (c)). By Theorem 7.2 any upward planar straight-line drawing of T^* with embedding $\mathcal{E}^*$ requires $\Omega(b^n)$ area, with b greater than 1. Hence the claimed lower bound follows.

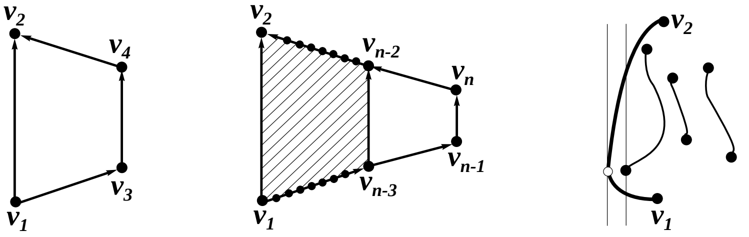
Although we believe that O_n doesn't admit any upward poly-line (possibly non-outerplanar) drawing in $o(n^2)$ area, we can more easily obtain a proof of Theorem 7.9 by considering the following family of directed outerplanar graphs.



(a) (b) (c)

Figure 7.9: (a) O_4. (b) O_{n+4}, defined in terms of O_n. (c) Outerplanar embedding of O_{12}. The thick lines show the subgraph T^* with embedding $\mathcal{E}^*$.

Let $\overline{O}_n$ be recursively defined as follows: (i) $\overline{O}_4$ has four vertices v_1, v_2, v_3, and v_4, an edge from v_1 to v_2, an edge from v_1 to v_3, an edge from v_3 to v_4, and an edge from v_4 to v_2 (see Fig. 7.10 (a)); (ii) $\overline{O}_n$ is obtained from $\overline{O}_{n-2}$ by adding to it two vertices v_{n-1} and v_n, an edge from v_{n-3} to v_{n-1}, an edge from v_{n-1} to v_n, and an edge from v_n to v_{n-2} (see Fig. 7.10 (b)).



(a) (b) (c)

Figure 7.10: (a) $\overline{O}_4$. (b) $\overline{O}_n$, defined in terms of $\overline{O}_{n-2}$. (c) Edge (v_1, v_2) touches a new vertical line with respect to those that intersect Γ_{i-1}.

In the following we show that $\overline{O}_n$ requires $\Omega(n^2)$ area in any upward polyline drawing. First, notice that $\overline{O}_n$ contains an n-vertex directed path $(v_1, v_3, \ldots, v_{n-3}, v_{n-1}, v_n, v_{n-2}, \ldots, v_4, v_2)$ as a subgraph, hence it has height at least n in any upward drawing.

Consider the set S of edges (v_{2i-1}, v_{2i}) of $\overline{O}_n$, with $i = 1, \ldots, n/2$. We prove by induction that $n/2$ width is required by any poly-line drawing of the edges in S in which (i) vertices and bends have integer coordinates, (ii) the y-coordinate of a vertex v_{2i} (of a vertex v_{2i-1}) is greater than (resp. is less than) the y-coordinate of a vertex v_{2i+2} (resp. of a vertex v_{2i+1}), for $i = 1, \ldots, n/2 - 1$, and (iii) all the edges (v_{2j-1}, v_{2j}) are contained in the same of the two vertically bounded regions delimited by edge (v_{2i-1}, v_{2i}) and by the horizontal lines through v_{2i-1} and through v_{2i}, for $i, j = 1, 2, \ldots, n/2$ and $j > i$ (i.e., all the edges (v_{2j-1}, v_{2j}) are on the same side of edge (v_{2i-1}, v_{2i})). Notice that in any planar upward grid drawing Γ_n of $\overline{O}_n$ the edges of S satisfy such properties. Namely, property (i) is satisfied by definition, property (ii) is satisfied by the upwardness of Γ_n, and property (iii) is satisfied by the planarity of Γ_n.

If $|S| = 1$, then the only edge of S requires one vertical line to be drawn. Now suppose, by induction, that if S has $n/2 - 1$ edges, then they require $n/2 - 1$ width in any poly-line drawing satisfying properties (i)–(iii). Consider any planar upward drawing Γ of a set S with $n/2$ edges. Denote by Γ_{-1} the sub-drawing of Γ obtained by not considering edge (v_1, v_2). If the width of Γ_{-1} is greater or equal than $n/2$, then the thesis follows. If the width of Γ_{-1} is equal to $n/2 - 1$, then there is a bend or a vertex on both the left and the right side of the bounding box, otherwise the width would be less or equal than $n/2 - 2$. To have all the other edges on the same side, the drawing of edge (v_1, v_2) must touch with a bend or a vertex at least one vertical line not intersecting Γ_{-1} (see Fig. 7.10 (c)), hence the thesis follows.

7.6 Conclusions and Open Problems

In this chapter we have studied the area requirements of upward drawings of directed trees and of several classes of directed acyclic graphs that frequently arise in theory and in practice.

We provided tight bounds on the area requirements of straight-/poly-line order-/non-order-preserving upward drawings of general directed trees and of several families of directed trees. However, the following problem is still open:

Open Problem 7.1 *Which are the asymptotic bounds for the area requirements of upward straight/poly-line order/non order-preserving drawings of directed complete and balanced trees?*

Concerning directed bipartite graphs, we have shown an exponential area lower bound for straight-line upward drawings. An interesting subclass of the directed bipartite graphs was defined in [DLR90] and called *bipartite directed acyclic graphs*. Such graphs are those directed acyclic graphs having a vertex set partitioned into two subsets V_1 and V_2 with each edge directed *from a vertex of* V_1 *to a vertex of* V_2.

Open Problem 7.2 *Which are the asymptotic bounds for the area requirements of upward drawings of bipartite directed acyclic graphs?*

Further, we have shown an outerplanar graph requiring exponential area in any straight-line *outerplanar* upward drawing. However, when considering *non-outerplanar* drawings, one could obtain better area bounds, so we ask:

Open Problem 7.3 *Which are the asymptotic bounds for the area requirements of upward straight-line (possibly non-outerplanar) drawings of directed outerplanar graphs?*

Chapter 8

Straight-line Drawings of Minimum Spanning Trees

In this chapter[1] we consider straight-line embeddings of minimum spanning trees in small area. We show that trees whose degree is bounded by four can be embedded as minimum spanning trees in polynomial area, namely in $O(n^{21.252})$ area. Better bounds are obtained for complete binary trees, binary trees, and complete ternary trees.

8.1 Introduction

In this chapter we study small-area drawings of a type of proximity graphs, namely minimum spanning trees, that have attracted intense research efforts and that have many applications in several fields of Theoretical Computer Science. As an example, minimum spanning trees are widely used in the field of sensor networks, namely their topologies guarantee total connection between the nodes of a network, while minimizing the total energy consumption of the sensors (see, e.g., [CLJ06]).

A *minimum spanning tree* of a set P of points in the plane is defined as a tree having a vertex for each point of P and having minimum total edge length.

[1] The contents of this chapter are a joint work with Michael Kaufmann, appeared in [Kau07, FK08] and submitted to journal. Thanks to the organizers and participants of the Workshop on Graph Drawing and Computational Geometry held in Bertinoro, Italy, where the research for this work started. The results on complete binary trees have been achieved together with Roberto Tamassia. Thanks also to Markus Geyer and Barbara Pampel for many critical remarks and suggestions.

The minimum spanning reflects certain proximity relations in a set of points in the plane, as shown by the following nice property, that is throughout the chapter cited as the *MST condition*.

Property 8.1 *A straight-line drawing Γ of a tree T is an embedding of T as a minimum spanning tree if and only if, for each pair of non-adjacent nodes u and v of T, their Euclidean distance in Γ is greater or equal than the length of each edge in the path connecting u and v in T.*

Given a set P of n points in the plane, it is well-known that the minimum spanning tree of P can be computed in optimal $\Theta(n \log n)$ time, however the computation of a minimum spanning tree subject to further constraints is often required. The boundedness of the degree of the nodes of the tree is a natural constraint to consider, since having high-degree nodes is in many ways undesiderable. It is well-known that every set of points in the plane has a minimum spanning tree with maximum degree 5 [MS92]. If the maximum degree of the nodes is constrained to be bounded by 2 or 3, then computing a minimum spanning tree is $\mathcal{NP}$-hard [GJ79, PV84] (the complexity status of the same problem is still unknown if the degree of the tree is bounded by 4). However, a polynomial-time approximation scheme is known if the maximum degree of the tree is required to be at most 2 [Aro98, Mit99], an $O(n^{\log^c n})$-time $(1+\epsilon)$-approximation algorithm [AC04] and a polynomial-time 1.402-approximation algorithm [Cha04] are known if the maximum degree of the tree is required to be at most 3, and a polynomial-time 1.143-approximation algorithm [Cha04] is known if the maximum degree of the tree is required to be at most 4.

Consider a tree T. Does T admit an *MST embedding*, i.e., a straight-line drawing in which the minimum spanning tree of the points where the vertices of T are placed at coincides with T? Monma and Suri [MS92] provided an algorithm to construct an MST embedding of any tree of maximum degree 5 and proved that any tree having a node of degree at least 7 does not admit an MST embedding. Eades and Whitesides [EW96b] filled the gap in Monma and Suri's results, by proving that deciding whether an MST embedding exists for a given tree of maximum degree 6 is $\mathcal{NP}$-hard.

Extensions to higher dimensions have been performed by Di Battista and Liotta [LB95], as well as by King [Kin06]. In the former paper, the authors proved that trees with maximum degree 9 can be embedded as MSTs in the three-dimensional space; in the latter paper, it is proved that every tree of maximum degree 10 admits an MST embedding in the three-dimensional space.

It is also known that no tree having a vertex of degree at least 13 admits an MST embedding in the three-dimensional space [Lee56].

Monma and Suri's proof that every tree of maximum degree five admits an MST embedding in the plane is a strong combinatorial result. However, their algorithm for constructing MST embeddings of trees turns out to be useless in practice, since the constructed drawings require an area of $O(2^{k^2}) \times O(2^{k^2})$ for trees of height k (hence, in the worst case the area requirement of the drawings is doubly-exponential in the number of nodes of the tree). Notice that the algorithm of Monma and Suri does not give a polynomial area bound even for *complete binary trees*, namely the algorithm provides an $O(n^{\log n})$ area bound in such a case. However, Monma and Suri conjectured that there exist trees of maximum degree 5 that require $c^n \times c^n$ area in *any* MST embedding, for some constant $c > 1$. The problem of determining whether or not the area upper bound for MST embeddings of trees can be improved to polynomial is reported also in [EW96b].

In this chapter, we concentrate on the area requirements for MST embeddings of trees in the plane. In particular we present the following area bounds: (i) Complete binary trees admit MST embeddings in $O(n^{4.3})$ area; (ii) binary trees admit MST embeddings in $O(n^{11.387})$ area; (iii) complete ternary trees admit MST embeddings in $O(n^{3.73})$ area; and (iv) ternary trees admit MST embeddings in $O(n^{21.252})$ area.

Table 8.1 summarizes the best known area bounds for straight-line drawings of trees.

	Upper bound	*Ref.*
Complete Degree 3	$O(n^{4.3})$	Th. 8.1
Degree 3	$O(n^{11.387})$	Th. 8.2
Complete Degree 4	$O(n^{3.73})$	Th. 8.3
Degree 4	$O(n^{21.252})$	Th. 8.4
Complete Degree 5	$O(n^{\log n})$	[MS92]
Degree 5	$O(2^{n^2})$	[MS92]

Table 8.1: Summary of the best known area bounds for MST-embeddings of trees.

The rest of the chapter is organized as follows. In Sect. 8.2 we show how to construct MST embeddings of complete binary trees; in Sect. 8.3 we show how to construct MST embeddings of arbitrary binary trees; in Sect. 8.4 we show how to construct MST embeddings of complete ternary trees; in Sect. 8.5 we show how to construct MST embeddings of arbitrary ternary trees; finally, in Sect. 8.6 we conclude and present some open problems.

We notice that a polynomial area bound for arbitrary ternary trees implies polynomial area bounds for complete binary trees, for arbitrary binary trees, and for complete ternary trees, that are all subclasses of arbitrary ternary trees. However, we still present algorithms for constructing MST embeddings of complete binary trees, of arbitrary binary trees, and of complete ternary trees, motivated both by the simplicity of the corresponding algorithms, and by the better area bounds that we can achieve in such cases. Notice also that we do not strive for the best polynomial bounds but try to keep the techniques and the analysis as simple as possible. Nevertheless, we achieve the first polynomial area bounds drastically improving from the previous exponential ones.

We also notice that throughout this chapter we deal with a model for graph drawing that is slightly different with respect to the one used for obtaining the previous results about small-area drawings of graphs. In fact, we do not force vertices to be on a grid (this choice is quite usual when dealing with proximity drawings); however, we ensure that every pair of vertices in the drawing are at distance at least one unit. For this purpose, it sufficient to ensure that every pair of *adjacent* vertices in the drawing are at distance at least one unit. In fact, if all the edges have length at least one unit, two non-adjacent vertices u and v cannot be closer than one unit distance, otherwise the path connecting u and v would contain edges longer than the Euclidean distance between u and v, hence violating the MST condition.

8.2 MST Embeddings of Complete Binary Trees

In this section we deal with the construction of MST embeddings of complete binary trees.

Let T be a complete binary tree with n nodes and let $n = 2^k - 1$, for some integer k. Tree T consists of a root r and of two subtrees T_1 and T_2 rooted at the children r_1 and r_2 of r, respectively. Each of T_1 and T_2 has size less than $n/2$. We recursively embed T_1 and T_2 into two equal isosceles right triangles Δ_1 and Δ_2, respectively, so that the root of a subtree T_i is placed at the vertex of Δ_i incident to the 90-degree angle.

When T has only one node, such a node is placed at the vertex incident to the 90 degrees angle of an isosceles right triangle Δ having sides of length one.

When T has more than one node, we place Δ_1 and Δ_2 with their hypotenuses on the same horizontal line, at distance d from each other, where d is a value that will be chosen later. Let L denote the length of a side of Δ_1 and Δ_2. We place r at the intersection of the perpendicular lines on which a side of Δ_1 and a side of Δ_2 lie. The whole drawing is contained inside an isosceles right triangle Δ having sides of length $(c+1)L$, where c is a constant that will be determined later. Observe that r is placed at the vertex of Δ incident to the 90-degree angle. See Fig. 8.1.

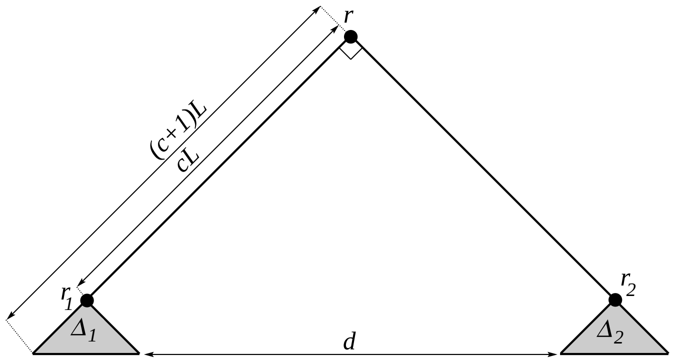


Figure 8.1: The recursive construction of an MST embedding of a complete binary tree.

We prove that the constructed drawing is an MST embedding of T, for some value of c. Inductively assume that the drawings of subtrees T_1 and T_2 are MST embeddings. Then, we have only to prove that each straight-line segment connecting a node w_1 in T_1 and a node w_2 in T_2 is longer than each edge of the path connecting w_1 and w_2 in T. By construction, the distance between w_1 and w_2 is at least d. The edges belonging to the path connecting w_1 and w_2 in T have length at most $\max\{\sqrt{2}L, cL\}$, namely all such edges are contained inside Δ_1 and Δ_2, but for (r, r_1) and (r, r_2), that by construction have length cL. Observe that, by construction, $d = \sqrt{2}(c-1)L$. Hence, as long as $c \geq \sqrt{2}/(\sqrt{2}-1)$, d is greater or equal than both cL and $\sqrt{2}L$, so the constructed drawing is an MST embedding of T.

We now compute the area of the constructed drawing, which is bounded by the area of Δ. Observe that each edge of the drawing has length at least one. Denote by $S(n)$ the length of the side of Δ, when the input is a complete binary tree with n nodes. We get: $S(n) = (c+1)S(\frac{n}{2}) = \left(\frac{2\sqrt{2}-1}{\sqrt{2}-1}\right)^{\log_2 n} = n^{\log_2 \frac{2\sqrt{2}-1}{\sqrt{2}-1}} \leq n^{\log_2 4.415} \leq n^{2.15}$. Since the area of Δ is asymptotically the square of its side, we obtain the following:

Theorem 8.1 *A complete binary tree with n vertices admits an MST embedding in $O(n^{4.3})$ area.*

8.3 MST Embeddings of Arbitrary Binary Trees

In this section we present an algorithm to construct MST embeddings of arbitrary binary trees.

Overall strategy. Assume that the input binary tree T is rooted at any node r of degree at most two. Select a spine $P = (r = v_1, v_2, v_3, \cdots, v_k)$ in T. Remove the spine from the tree, disconnecting the tree into several subtrees. Recursively draw the disconnected subtrees and place a drawing of the spine together with the drawings of the subtrees, obtaining a drawing of the whole tree.

Choice of the spine. The choice of P is done as follows. The first node v_1 of P is r. For each $1 \leq i < k$, node v_{i+1} is defined as the root of the largest of the two subtrees of v_i. Observe that each subtree of P has at most $n/2$ nodes.

The shape of the subtrees. Denote by T_i the subtree rooted at the child t_i of v_i that does not belong to P. We recursively draw the subtrees T_i of P inside isosceles right triangles Δ_i, for $1 \leq i \leq k-1$. The whole spine together with the drawing of the subtrees of P will be placed inside a larger isosceles right triangle Δ. The root of each subtree T_i is placed on the midpoint of the hypotenuse of Δ_i. Denote by L_i the length of the hypotenuse of Δ_i.

Drawing the spine and the subtrees together. Let $e_i = (v_i, v_{i+1})$, for $1 \leq i < k$. We draw P in a *zig-zag* way, with constant angles of 120 degrees between two consecutive edges e_i and e_{i+1}. The length of edges e_i will be determined later.

Consider vertex v_i. Opposite to the 120 degree angle, we have an angle of 240 degrees, which we partition into four consecutive wedges W_i^1, W_i^2, W_i^3, and W_i^4 of 90, 30, 30, and 90 degrees, respectively, such that W_i^1 is the wedge closer to vertex v_{i-1}. See Fig. 8.2. We place Δ_i inside W_i^3 as follows. Consider the line l_i through v_i bisecting W_i^3. Vertex t_i is placed on l_i and triangle Δ_i is placed inside W_i^3 so that the hypotenuse of Δ_i is perpendicular to l_i, and so that the endvertices of the hypotenuse of Δ_i lie on the semi-axes delimiting W_i^3. See Fig. 8.3.

Notice that, for vertex v_1 (and for vertex v_k), wedges W_i^1, W_i^2, W_i^3, and W_i^4 are not well-defined, since only one edge e_1 of P is incident to v_1. However,

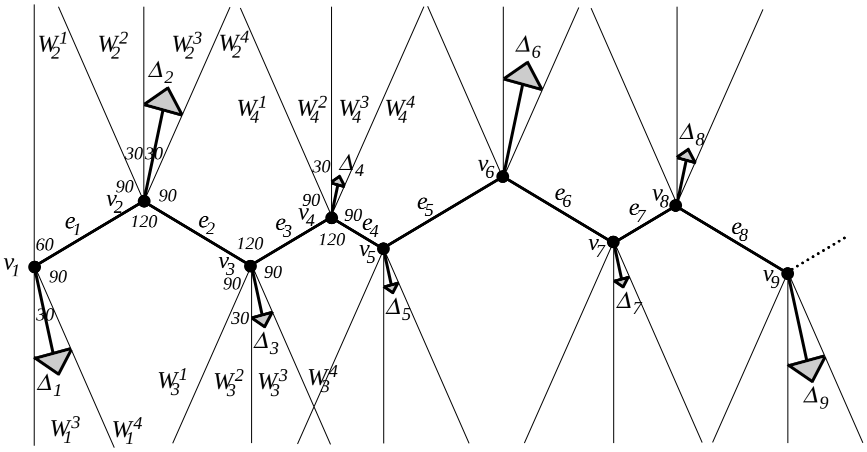


Figure 8.2: The recursive construction of an MST embedding of an arbitrary binary tree.

it is not difficult to extend the above definition of wedges W_i^1, W_i^2, W_i^3, and W_i^4 to the case in which $i = 1$, by considering a dummy edge (v_0, v_1) that has an angle of 120 degrees with edge (v_1, v_2), and defining the wedges incident to v_1 as for the other vertices of P.

Choosing the length of edges $\mathbf{e}_i$. We set:

$$len(e_i) = max\{cL_i, cL_{i+1}\},$$

where c is a constant greater than one to be determined later. In order to have positive lengths for all the edges, we set $len(e_i) = 1$, for all the edges e_i where none of subtrees T_i and T_{i+1} exists.

The isosceles right triangle Δ is defined as the smallest isosceles right triangle containing the whole drawing, having r as midpoint of the hypotenuse, and having the hypotenuse forming angles of 120, 60, and 180 degrees with edge (v_1, v_2). In the following we suppose, for clarity of exposition, that the hypotenuse of Δ is vertical, and that P is contained in the half-plane to the right of the line through the hypotenuse. If a subtree T_i has only one node, Δ is defined as the isosceles right triangle having r as midpoint of the hypotenuse, and having the hypotenuse such that $L_i = 1$.

The drawing satisfies the MST condition. We use induction to show that

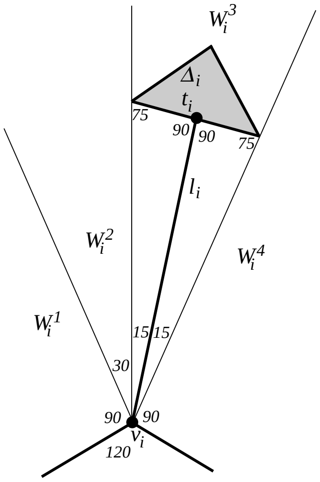


Figure 8.3: A closer look to the construction of an MST embedding of an arbitrary binary tree.

every pair of vertices in the drawing satisfies the MST condition. If the tree has only one node, then there is nothing to prove. Otherwise, inductively suppose that each pair of nodes in the drawing of each subtree T_i satisfies the MST condition. Then, we prove that each pair of nodes in the whole drawing satisfies the MST condition.

The only pairs of nodes for which the MST condition is not trivially satisfied, are: (i) node v_i and any node in T_i, for $i = 1, 2, \cdots, k-1$, (ii) node v_i and any node in T_{i-1}, for $i = 2, 3, \cdots, k$, (iii) node v_i and any node in T_{i+1}, for $i = 1, 2, \cdots, k-2$, and (iv) any node in $T_{i-1} \cup \{v_{i-1}\}$ and any node in $T_{i+1} \cup \{v_{i+1}\}$, for $i = 2, 3, \cdots, k-2$.

(i) Consider node v_i and any node w_i in T_i, for any $i = 1, 2, \cdots, k-1$. We prove that all the edges in the path from v_i to w_i are shorter than segment $\overline{v_i w_i}$. Each edge of such a path belonging to T_i has length at most L_i. The length of edge (v_i, t_i) is equal to $L_i/(2 \cdot \tan(15)) \geq 1.866 L_i$. Hence, (v_i, t_i) is the longest edge of the path connecting v_i and w_i. However, $\overline{v_i w_i}$ is longer than (v_i, t_i), since w_i is contained inside Δ_i, whose closest point to v_i is t_i.

(ii) For any $i = 2, 3, \cdots, k-1$, consider a node v_i and any node w_{i-1} in T_{i-1}, and suppose that the pair (v_i, w_{i-1}) of vertices does not satisfy the MST condition. As in the previous case each edge of such a path belonging also to T_{i-1} has length at most L_{i-1}. Further, edge (v_{i-1}, t_{i-1}) has length $L_i/(2 \cdot \tan(15)) \leq 1.867L_i$, and edge (v_{i-1}, v_i) has length at least cL_{i-1}. It follows that, as long as $c \geq 1.867$, edge (v_{i-1}, v_i) is the longest edge in the path connecting v_i and w_{i-1}. However, consider triangle (v_i, v_{i-1}, w_{i-1}). By construction, angle $v_i\widehat{v}_{i-1}w_{i-1}$ contains wedge W_i^4 and hence it is greater or equal than 90 degrees. Segment $\overline{v_iw_i}$ is opposite to $v_i\widehat{v}_{i-1}w_{i-1}$ and hence is the longest side of such a triangle. It follows that $\overline{v_iw_i}$ is longer than (v_{i-1}, v_i).

(iii) For any $i = 1, 2, \cdots, k-2$, it can be proved analogously to the previous case that the MST of the points of the drawing cannot contain an edge (v_i, w_{i+1}), for any node w_{i+1} in T_{i+1}.

(iv) Consider any node w_{i-1} in $T_{i-1} \cup \{v_{i-1}\}$ and any node w_{i+1} in $T_{i+1} \cup \{v_{i+1}\}$, for $i = 2, 3, \cdots, k-2$. The path P_{i-1}^{i+1} connecting w_{i-1} and w_{i+1} in T contains edges e_{i-1} and e_i. All the edges of P_{i-1}^{i+1} belonging to T_{i-1} or to T_{i+1} are contained inside Δ_{i-1} or Δ_{i+1}, respectively, and hence their length is at most the maximum between L_{i-1} and L_{i+1}. Further, the length of edge $\overline{v_{i-1}t_{i-1}}$ is $L_{i-1}/(2 \cdot \tan(15)) \leq 1.867L_{i-1}$. Analogously, the length of edge $\overline{v_{i+1}t_{i+1}}$ is at most $1.867L_{i+1}$. Hence, the length of each edge in P_{i-1}^{i+1} is less or equal than $\max\{1.867L_{i-1}, 1.867L_{i+1}, len(e_{i-1}), len(e_i)\}$. Observe that, by construction, $len(e_{i-1}) \geq cL_{i-1}$, and that $len(e_i) \geq cL_{i+1}$. Hence, as long as $c \geq 1.867$, one edge between e_i and e_{i+1} is the longest edge in P_{i-1}^{i+1}, and we have only to prove that the distance between w_{i-1} and w_{i+1} is greater than $\max\{len(e_{i-1}), len(e_i)\}$. In the following, refer to Fig. 8.4.

Consider line $l_{i-1}^{3,4}$ separating wedges W_{i-1}^3 and W_{i-1}^4, and consider line $l_i^{1,2}$ separating wedges W_i^1 and W_i^2. By construction such lines are parallel. Further, T_{i-1} is contained in the half-plane delimited by $l_{i-1}^{3,4}$ and not containing $l_i^{1,2}$. Notice that the distance between $l_{i-1}^{3,4}$ and $l_i^{1,2}$ is exactly $len(e_{i-1})$. We claim that, for a suitable constant c, T_{i+1} is entirely contained in the half-plane delimited by $l_i^{1,2}$ and not containing $l_{i-1}^{3,4}$. The claim clearly implies that the distance between w_{i-1} and w_{i+1} is greater or equal than $len(e_{i-1})$.

Let v_{i+1}^C be the vertex of Δ_{i+1} on the line l_{i+1}^C separating wedges W_{i+1}^2 and W_{i+1}^3. By construction, T_{i+1} entirely lies in the half-plane that is delimited by the line with slope 60 degrees through v_{i+1}^C and that does not contain $l_{i-1}^{3,4}$ and $l_i^{1,2}$. Hence, we have only to prove that, for a suitable constant c, v_{i+1}^C is in the half-plane delimited by $l_i^{1,2}$ and not containing $l_{i-1}^{3,4}$.

The vertical distance between v_{i+1} and v_{i+1}^C is easily computed to be $L_{i+1}/(2 \cdot \sin(15))$. The vertical distance between v_{i+1} and the intersection point u_i^C of l_i^C and $l_i^{1,2}$ is exactly $len(e_i)$, since triangle (v_i, v_{i+1}, u_i^C) is an isosceles triangle

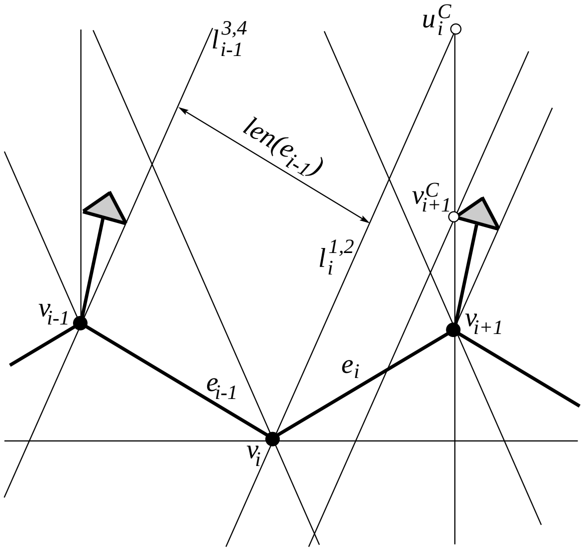


Figure 8.4: Illustration for the proof that the MST condition is satisfied for any node in $T_{i-1} \cup \{v_{i-1}\}$ and any node in $T_{i+1} \cup \{v_{i+1}\}$, for $i = 2, 3, \cdots, k-2$.

with catheti (v_i, v_{i+1}) and (v_{i+1}, u_i^C). It follows that $\overline{v_{i+1}u_i^C}$ is at least cL_{i+1}. Hence, v_{i+1}^C is in the half-plane delimited by $l_i^{1,2}$ and not containing $l_{i-1}^{3,4}$ as long as $cL_{i+1} \geq L_{i+1}/(2 \cdot \sin(15))$, i.e., as long as $c \geq 1.932$.

In analogous way, it can be proved that, as long as $c \geq 1.932$, the distance between w_{i-1} and w_{i+1} is greater than $len(e_i)$. Hence, as long as $c \geq 1.932$, the straight-line segment between w_{i-1} and w_{i+1} is longer than every edge in the path P_{i-1}^{i+1} connecting w_{i-1} and w_{i+1} in T, and hence it does not belong to the MST of the points of the drawing.

The length of **P**. We bound the length of P as a function of the lengths L_i's. Since $len(e_i) = max\{cL_i, cL_{i+1}\}$, and since $len(e_i) \geq 1$, for every $1 \leq i < k$, then $len(e_i) < cL_i + cL_{i+1}$. It follows that $\sum_{i=1}^{k-1} len(e_i) \leq 2c \sum_{i=1}^{k-1} L_i$.

The area of the drawing is polynomial. We now compute the length of $h(C)$, i.e., of the hypotenuse of an isosceles right triangle that contains the whole drawing, that has r as midpoint of its hypotenuse, and that has the hypotenuse forming angles of 120, 60, and 180 degrees with edge (v_1, v_2). In the following refer to Fig. 8.5. Notice that the length of the longest edge of the drawing is at most equal to $h(C)$, while the length of the shortest edge of the drawing is at least 1, by construction.

We first notice that the drawing of P (without the drawing of subtrees $T_i's$) is contained inside an equilateral triangle Δ_e that has r as a vertex and such that the two sides incident to r have length equal $2c\sum_{i=1}^{k-1} L_i$ and form angles of 60 degrees with $h(C)$. In fact, the length of P is at most $2c\sum_{i=1}^{k-1} L_i$, and, since each edge of P forms an angle of 30 degrees with a horizontal line, the horizontal extension of P is at most $2c\sum_{i=1}^{k-1} L_i \cdot cos(30)$.

Consider the smallest isosceles right triangle Δ^* that contains Δ_e completely, that has r as midpoint of its hypotenuse, and that has the hypotenuse forming angles of 120, 60, and 180 degrees with edge (v_1, v_2). Easy trigonometric calculations show that the hypotenuse of Δ^* has length at most $2(\cos(60)+\sin(60))(2c\sum_{i=1}^{k-1} L_i) = 5.46411c\sum_{i=1}^{k-1} L_i$.

Since edge (v_i, t_i) has length at most $L_i/(2 \cdot \tan(15)) \leq 1.867L_i$ and since all the points of Δ_i are at distance at most $L_i/2$ from t_i, then no point of Δ_i is at distance greater than $2.367L_i$ from v_i. Consider the smallest isosceles right triangle Δ that contains Δ^*, that has r as midpoint of its hypotenuse, that has the hypotenuse forming angles of 120, 60, and 180 degrees with edge (v_1, v_2), and such that every point on one of its catheti has distance at least $2.367\sum_{i=1}^{k-1} L_i$ from any point of Δ^*. It is easy to see that Δ contains the whole drawing, namely it contains P since it contains Δ^*, and it contains each subtree T_i, since T_i can stick outside Δ^* by at most $L_i/2 + 1.867L_i = 2.367L_i \leq 2.367\sum_{i=1}^{k-1} L_i$. Notice that the hypotenuse of Δ has length at most $5.46411c\sum_{i=1}^{k-1} L_i + 2(2.367\sqrt{2}\sum_{i=1}^{k-1} L_i)$. By choosing $c = 1.932$, the drawing of T is an MST embedding, and the length of the hypotenuse of smallest right isosceles triangle containing the drawing is bounded by $5.46411 \cdot 1.932\sum_{i=1}^{k-1} L_i + 2(2.367\sqrt{2}\sum_{i=1}^{k-1}) = 17.246\sum_{i=1}^{k-1} L_i$.

Lemma 8.1 *The length of $h(C)$ is at most $17.246\sum_{i=1}^{k-1} L_i$.*

Let $\alpha = 17.246$. Now, we express $h(C)$ as a function of the number of nodes of the tree. Denoting by $h(n)$ the maximum length of $h(C)$ when the input tree has n nodes, we inductively prove that $h(n) \leq n^{\log_2(3\alpha)}$. By Lemma 8.1, we get $h(n) \leq \alpha\sum_{i=1}^{k-1} h(n_i)$, where n_i is the number of nodes in T_i. By inductive hypothesis we get $h(n) \leq \alpha\sum_{i=1}^{k-1} n_i^{\log_2(3\alpha)}$. Group the numbers n_i in at most three groups N_1, N_2, and N_3 such that $\sum_{n_i \in N_1} n_i \leq \frac{n}{2}$, $\sum_{n_i \in N_2} n_i \leq \frac{n}{2}$, and $\sum_{n_i \in N_3} n_i \leq \frac{n}{2}$. Notice that it is always possible to construct such groups,

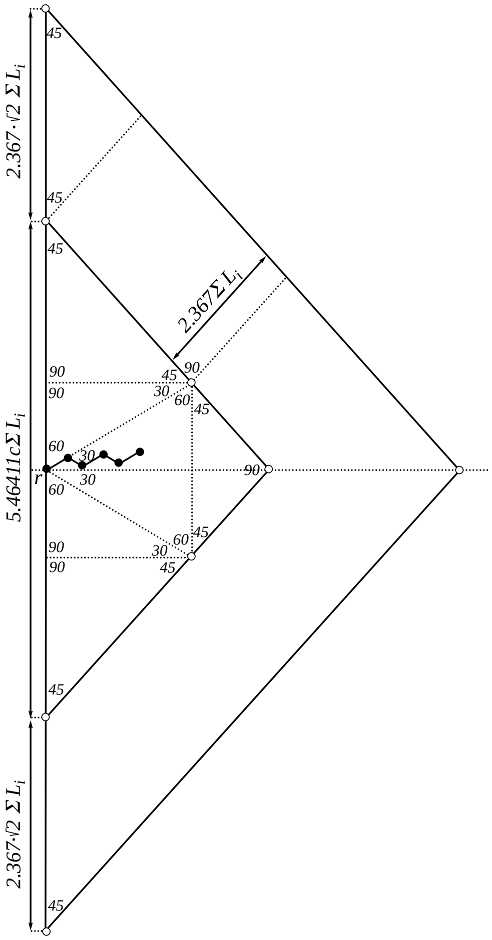


Figure 8.5: Bounding the constructed drawing with an isosceles right triangle.

namely start from groups $\{n_i\}$, each one containing a single value n_i, for $1 \leq i \leq k-1$. Since each subtree T_i has at most $n/2$ vertices, then $n_i \leq n/2$ and each starting group contains numbers adding up to at most $n/2$. Till there are more than three groups of numbers, consider any four groups of numbers. The numbers in the two groups that have minimal sum of their numbers add up to at most $n/2$ (otherwise the sum of the n_i's would be more than n). Hence, such groups can be joined to be the same group, hence decreasing the number of groups by one. Therefore, we have:

$$\begin{aligned}
h(n) &\leq \alpha \sum_{i=1}^{k-1} n_i^{\log_2(3\alpha)} = \\
&= \alpha \left(\sum_{n_i \in N_1} n_i^{\log_2(3\alpha)} + \sum_{n_i \in N_2} n_i^{\log_2(3\alpha)} + \sum_{n_i \in N_3} n_i^{\log_2(3\alpha)} \right) \leq \\
&\leq \alpha \left(\left(\sum_{n_i \in N_1} n_i \right)^{\log_2(3\alpha)} + \left(\sum_{n_i \in N_2} n_i \right)^{\log_2(3\alpha)} + \left(\sum_{n_i \in N_3} n_i \right)^{\log_2(3\alpha)} \right) \leq \\
&\leq \alpha \left(\left(\frac{n}{2}\right)^{\log_2(3\alpha)} + \left(\frac{n}{2}\right)^{\log_2(3\alpha)} + \left(\frac{n}{2}\right)^{\log_2(3\alpha)} \right) \leq 3\alpha \left(\frac{n}{2}\right)^{\log_2(3\alpha)} = \\
&= 3\alpha \frac{n^{\log_2(3\alpha)}}{2^{\log_2(3\alpha)}} = 3\alpha \frac{n^{\log_2(3\alpha)}}{3\alpha} = n^{\log_2(3\alpha)},
\end{aligned}$$

in which we used $\sum(n_i^k) \leq (\sum n_i)^k$. Hence, the inductive hypothesis is verified, and we can conclude that $h(n) \leq n^{\log_2 51.738} = O(n^{5.6932})$.

Finally, since the area of the drawing is the square of the length of its side, we get the following:

Theorem 8.2 *Every binary tree with n vertices admits an MST drawing in $O(n^{11.387})$ area.*

8.4 MST Embeddings of Complete Ternary Trees

In this section we deal with the construction of MST embeddings of complete ternary trees.

Let T be a complete ternary tree with n nodes and let $n = \frac{3^k-1}{2}$, for some integer k. Tree T consists of a root r and of three subtrees T_1, T_2, and T_3 rooted at the children r_1, r_2, and r_3 of r, respectively. Each of T_1, T_2, and T_3 has size less than $n/3$. We recursively embed T_1, T_2, and T_3 into three equal isosceles right triangles Δ_1, Δ_2, and Δ_3, respectively, so that the root of a subtree T_i is placed at the midpoint of the hypotenuse of Δ_i. In the base case, i.e., when T has only one node r, assume that r is placed at the midpoint of the hypotenuse of an isosceles right triangle Δ having the hypotenuse of length 1.

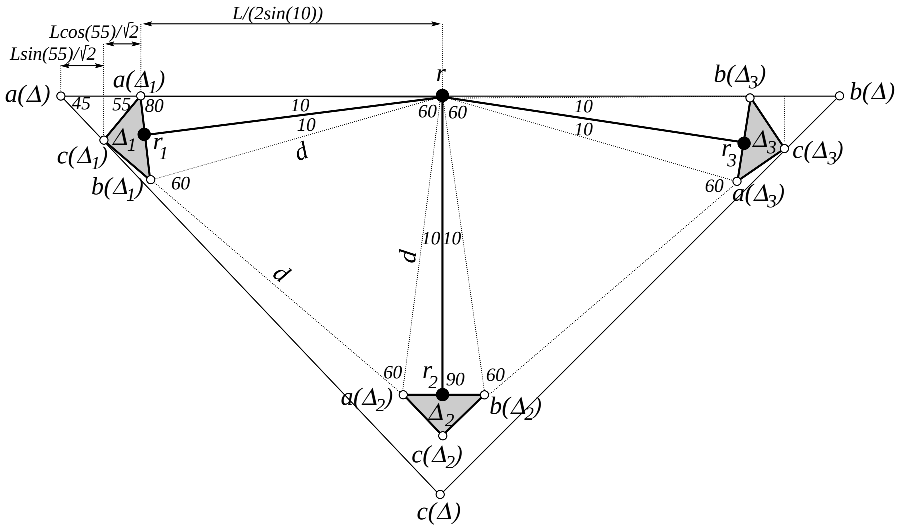


Figure 8.6: Bounding the constructed drawing with an isosceles right triangle.

In the inductive case we construct a drawing of T inside an isosceles right triangle Δ as follows. Refer to Fig. 8.6. Let L denote the length of the hypotenuse of Δ_1, Δ_2, and Δ_3. Denote also by $a(\Delta_i)$, $b(\Delta_i)$, and $c(\Delta_i)$ the vertices of Δ_i, for $i = 1, 2, 3$, so that $a(\Delta_i)$ and $b(\Delta_i)$ are the end-vertices of the hypotenuse of Δ_i. Place r in the plane. Δ_2 is placed with its hypotenuse lying on a horizontal line, so that the segment connecting r and r_2 is perpendicular to the line through $a(\Delta_2)$ and $b(\Delta_2)$, and so that angles $r_2\hat{r}a(\Delta_2)$ and $r_2\hat{r}b(\Delta_2)$ are both of 10 degrees. Denote by d the distance between r and $a(\Delta_2)$. Δ_1 is placed with $a(\Delta_1)$ on the horizontal line through r, with $b(\Delta_1)$ at distance d from both r and $a(\Delta_2)$, so that angles $r_1\hat{r}a(\Delta_1)$ and $r_1\hat{r}b(\Delta_1)$ are both of 10 degrees, and so that segment $\overline{rr_1}$ is perpendicular to the line through $a(\Delta_1)$ and $b(\Delta_1)$. Δ_3 is placed in the plane symmetrically to Δ_1 with respect to a vertical line through r. The whole drawing is contained inside an isosceles right triangle Δ with hypotenuse that lies on a horizontal line and that has a length to be computed later.

We prove that the constructed drawing is an MST embedding of T. Inductively assume that the drawings of subtrees T_1, T_2, and T_3 are MST embeddings. We prove that each straight-line segment connecting a node w_1 in T_1 and a node w_2 in T_2 is longer than each edge of the path connecting w_1 and w_2 in T. By construction, the distance between w_1 and w_2 is at least d. The edges belonging to the path connecting w_1 and w_2 in T have length that is at most $\max\{L, L/(2 \cdot tan(10))\} = \max\{L, 2.836L\} = 2.836L$, namely all such edges are contained inside Δ_1 and Δ_2, but for (r, r_1) and (r, r_2), that by construction have length at most $L/(2 \cdot tan(10))$. Observe that, by construction, $d = L/(2 \cdot sin(10)) > 2.879L$. Hence, the distance between each pair of nodes w_1 and w_2 in T_1 and in T_2, respectively, satisfies the MST condition. It can be proved analogously that each pair of nodes w_2 and w_3 in T_2 and in T_3, respectively, satisfies the MST condition. Further, the MST condition is trivially satisfied for each pair of nodes w_1 and w_3 in T_1 and in T_3, respectively.

We now compute the area of the constructed drawing. Namely, we bound the constructed drawing by an isosceles right triangle Δ such that r is placed at the midpoint of the hypotenuse of Δ. Consider the line $l(\Delta_1)$ with slope -45 degrees passing through $c(\Delta_1)$. We claim that all the drawing is contained in the half-plane to the right of $l(\Delta_1)$. The claim is proved by the following two considerations: 1) Δ_1 is contained in the half-plane to the right of $l(\Delta_1)$, namely the slope of the segment connecting $c(\Delta_1)$ and $b(\Delta_1)$ is -35 degrees; 2) Δ_2 is contained in the half-plane to the right of $l(\Delta_1)$, namely the distance between r and $c(\Delta_2)$ is easily computed to be $L/(2 \cdot \tan(10)) + L/2 < 3.34L$, which is less than the distance between r and the intersection point of $l(\Delta_1)$ and the horizontal line through r. In fact, such a distance is equal to $\frac{L}{2 \cdot \sin(10)} + \frac{L \cdot \cos(55)}{\sqrt{2}} + \frac{L \cdot \sin(55)}{\sqrt{2}} > 3.864L$.

The length of the hypotenuse of Δ is twice the length of segment $\overline{ra(\Delta_1)}$, hence the hypotenuse of Δ has length less or equal than $7.7284L$. Observe that each edge of the drawing has length at least 1. Denote by $h(n)$ the length of the hypotenuse of Δ. We get: $h(n) \leq 7.7284h(\frac{n}{3}) \leq 7.7284^{\log_3 n} = n^{\log_3 7.7284} \leq n^{1.862}$. Since the area of Δ is asymptotically the square of its side, we obtain the following:

Theorem 8.3 *A complete ternary tree with* n *vertices admits an MST embedding in* $O(n^{3.73})$ *area.*

8.5 MST Embeddings of Arbitrary Ternary Trees

In this section we present an algorithm to construct MST-embeddings of arbitrary ternary trees.

Overall strategy. Assume that the input ternary tree T is rooted at any node r of degree at most three. Analogously to the arbitrary binary tree case, select a spine $P = (r = v_1, v_2, v_3, \cdots, v_k)$ in T. Remove the spine from the tree, disconnecting the tree into several subtrees. Recursively draw the disconnected subtrees and place a drawing of the spine together with the drawings of the subtrees, obtaining a drawing of the whole tree.

Choice of the spine. The choice of P is done as in the arbitrary binary trees case. The first node v_1 of P is r. For each $1 \leq i < k$, node v_{i+1} is defined as the root of the largest of the three subtrees of v_i. Observe that each subtree of P has at most $n/2$ nodes.

The shape of the subtrees. Denote by T_i^1 and T_i^2 the subtrees rooted at the children t_i^1 and t_i^2 of v_i that do not belong to P, respectively. We recursively draw subtrees T_i^1 and T_i^2, for all $1 \leq i \leq k-1$, inside isosceles right triangles Δ_i^1 and Δ_i^2, respectively. For each $1 \leq i \leq k-1$, we scale up the drawing of the smallest between Δ_i^1 and Δ_i^2, so that the two isosceles right triangles are congruent. The whole spine together with the drawing of the subtrees of the nodes of P will be placed inside a larger isosceles right triangle Δ. The root of each subtree T_i^1 and T_2^1 is placed on the midpoint of the hypotenuse of Δ_i^1 and Δ_i^2, respectively. Denote by L_i the length of the hypotenuse of Δ_i^1 and Δ_i^2.

Drawing the spine and the subtrees together. Let $e_i = (v_i, v_{i+1})$, for $1 \leq i < k$. We draw P in a *zig-zag* way, with constant angles of 110 degrees between two consecutive edges e_i and e_{i+1}. See Fig. 8.7. The length of edges e_i will be determined later.

Consider vertex v_i. Opposite to the 110 degree angle, we have an angle of 250 degrees, which we partition into five consecutive wedges W_i^1, W_i^2, W_i^3, W_i^4, and W_i^5 of 90, 5, 60, 5, and 90 degrees, respectively, such that W_i^1 is the wedge closer to vertex v_{i-1}. We place Δ_i^1 inside W_i^2 and Δ_i^2 inside W_i^4 as follows. Consider the line l_i^2 through v_i bisecting W_i^2. Vertex t_i^1 is placed on l_i^2 and triangle Δ_i^1 is placed inside W_i^2 so that the hypotenuse of Δ_i^1 is perpendicular to l_i^2, and so that the endvertices of the hypotenuse of Δ_i^1 lie on the semi-axes delimiting W_i^2. Δ_i^2 is analogously placed inside W_i^4. See Fig. 8.8.

Notice that, for vertex v_1 (and for vertex v_k), wedges W_i^1, W_i^2, W_i^3, W_i^4, and W_i^5 are not well-defined, since only one edge e_1 of P is incident to v_1.

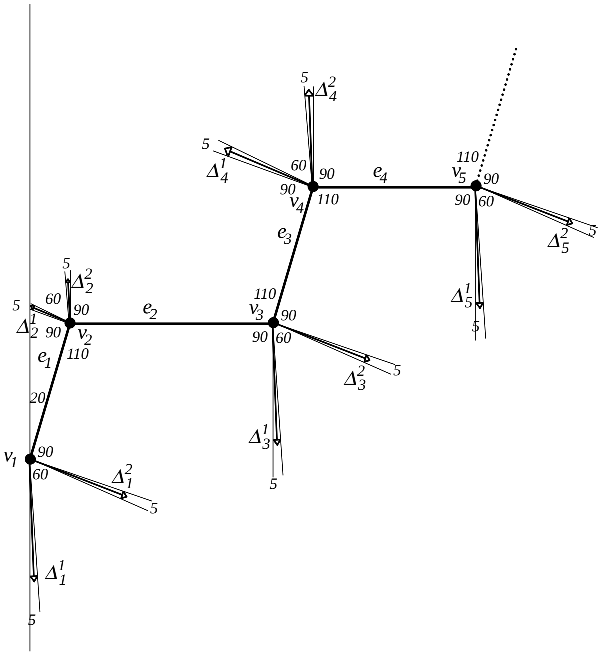


Figure 8.7: The recursive construction of an MST embedding of an arbitrary ternary tree. In order to improve readability, edges connecting the subtrees to the spine are longer than they should be (hence the actual drawing is not an MST embedding).

However, it is not difficult to extend the above definition of wedges W_i^1, W_i^2, W_i^3, W_i^4, and W_i^5 to the case in which $i = 1$, by considering a dummy edge (v_0, v_1) that has an angle of 110 degrees with edge (v_1, v_2), and defining the wedges incident to v_1 as for the other vertices of P.

Choosing the length of edges $\mathbf{e}_i$. As in the arbitrary binary tree case, we set:

$$len(e_i) = max\{cL_i, cL_{i+1}\},$$

where c is a constant to be determined later. In order to have length at least one for all the edges, we set $len(e_i) = 1$, for all the edges e_i where none of subtrees T_i^1, T_i^2, T_{i+1}^1, and T_{i+1}^2 exists.

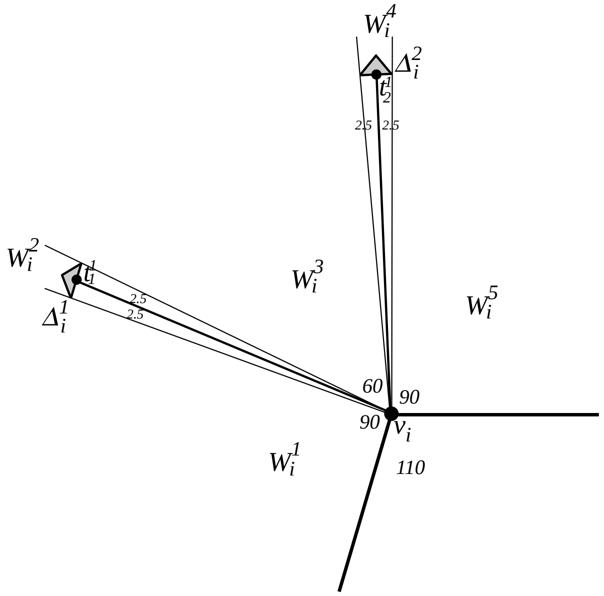


Figure 8.8: A closer look to the construction of an MST embedding of an arbitrary ternary tree.

The isosceles right triangle Δ is defined as the smallest isosceles right triangle containing the whole drawing, having r as midpoint of the hypotenuse, and having the hypotenuse forming angles of 160, 20, and 180 degrees with edge (v_1, v_2). In the following we suppose, for clarity of exposition, that the hypotenuse of Δ is vertical, and that P is contained in the half-plane to the right of the line through the hypotenuse. If a tree T has only one node, Δ is defined as the isosceles right triangle having r as midpoint of the hypotenuse, and having the hypotenuse of length 1.

The drawing satisfies the MST condition. We use induction to show that every pair of vertices in the drawing satisfies the MST condition. If the tree has only one node, then there is nothing to prove. Otherwise, inductively suppose that each pair of nodes in the drawing of each subtree T_i^1 and T_i^2 satisfies the MST condition. Then, we prove that each pair of nodes in the whole drawing satisfies the MST condition.

The only pairs of nodes for which the MST condition is not trivially satisfied, are: (i) node v_i and any node in T_i^1 or in T_i^2, for $i = 1, 2, \cdots, k-1$, (ii) any node in T_i^1 and any node in T_i^2, for $i = 1, 2, \cdots, k-1$, (iii) node v_i and any node in T_{i-1}^2, for $i = 2, 3, \cdots, k$, (iv) node v_i and any node in T_{i+1}^1, for $i = 1, 2, \cdots, k-2$, and (v) any node in $T_{i-1}^2 \cup \{v_{i-1}\}$ and any node in $T_{i+1}^1 \cup \{v_{i+1}\}$, for $i = 2, 3, \cdots, k-2$.

(i) Consider node v_i and any node w_i in T_i^1 (resp. in T_i^2), for any $i = 1, 2, \cdots, k-1$. We prove that all the edges in the path from v_i to w_i are shorter than segment $\overline{v_i w_i}$. The length of each edge of such a path belonging to T_i^1 (resp. to T_i^2) is at most L_i. The length of edge (v_i, t_i^1) (resp. edge (v_i, t_i^2)) is equal to $L_i/(2 \cdot \tan(2.5)) \geq 11.451 L_i$. Hence, (v_i, t_i^1) (resp. (v_i, t_i^2)) is the longest edge of the path connecting v_i and w_i. However, segment $\overline{v_i w_i}$ is longer than (v_i, t_i^1) (resp. than (v_i, t_i^2)), since w_i is contained inside Δ_i^1 (resp. inside Δ_i^2), whose closest point to v_i is t_i^1 (resp. t_i^2).

(ii) Consider any node n_i^1 in T_i^1 and any node n_i^2 in T_i^2, for any $i = 1, 2, \cdots, k-1$. We prove that all the edges in the path from n_i^1 to n_i^2 are shorter than segment $\overline{n_i^1 n_i^2}$. The length of each edge of such a path belonging also to T_i^1 or to T_i^2 is at most L_i. Further, edges (v_i, t_i^1) and (v_i, t_i^2) have length $L_i/(2 \cdot \tan(2.5)) \approx 11.451 L_i$. Hence, (v_i, t_i^1) and (v_i, t_i^2) are the longest edges in the path connecting n_i^1 and n_i^2. However, consider the intersection point $p(i_1)$ of Δ_i^1 and the line separating wedges W_i^2 and W_i^3, and consider the intersection point $p(i_2)$ of Δ_i^2 and the line separating wedges W_i^3 and W_i^4. The length of segment $\overline{n_i^1 n_i^2}$ is greater or equal than the length of segment $\overline{p(i_1)p(i_2)}$. By construction, triangle $(p(i_1), p(i_2), v_i)$ is equilateral, hence $\overline{p(i_1)p(i_2)}$ has the same length of segments $\overline{v_i p(i_1)}$ and $\overline{v_i p(i_2)}$, that is $L_i/(2 \cdot \sin(2.5)) \approx 11.462 L_i$, which is greater than $L_i/(2 \cdot \tan(2.5))$.

(iii) For any $i = 2, 3, \cdots, k-1$, consider a node v_i and any node w_{i-1} in T_{i-1}^2, and suppose that the pair (v_i, w_{i-1}) of vertices does not satisfy the MST condition. As in the previous case each edge of such a path belonging also to T_{i-1}^2 has length at most L_{i-1}. Further, edge (v_{i-1}, t_{i-1}^2) has length $L_{i_1}/(2 \cdot \tan(2.5)) \leq 11.452 L_{i_1}$, and edge (v_{i-1}, v_i) has length at least cL_{i-1}. It follows that, as long as $c \geq 11.452$, edge (v_{i-1}, v_i) is the longest edge in the path connecting v_i and w_{i-1}. However, consider triangle (v_i, v_{i-1}, w_{i-1}). By construction, angle $v_i \widehat{v}_{i-1} w_{i-1}$ contains wedge W_i^5 and hence it is greater or equal than 90 degrees. Segment $\overline{v_i w_{i-1}}$ is opposite to $v_i \widehat{v}_{i-1} w_{i-1}$ and hence is the longest side of such a triangle. It follows that $\overline{v_i w_i}$ is longer than (v_{i-1}, v_i).

(iv) For any $i = 1, 2, \cdots, k-2$, it can be proved analogously to the previous case that the MST of the points of the drawing cannot contain an edge (v_i, w_{i+1}), for any node w_{i+1} in T_{i+1}^1.

(v) Consider any node w_{i-1} in $T^2_{i-1} \cup \{v_{i-1}\}$ and any node w_{i+1} in $T^1_{i+1} \cup \{v_{i+1}\}$, for $i = 2, 3, \cdots, k-2$. The path P^{i+1}_{i-1} connecting w_{i-1} and w_{i+1} in T contains edges e_{i-1} and e_i. All the edges of P^{i+1}_{i-1} belonging to T^2_{i-1} or to T^1_{i+1} are contained inside Δ^2_{i-1} or Δ^1_{i+1}, respectively, and hence their length is at most the maximum between L_{i-1} and L_{i+1}. Further, the length of edge (v_{i-1}, t^2_{i-1}) is $L_{i-1}/(2 \cdot \tan(2.5)) \leq 11.452L_{i-1}$. Analogously, the length of edge (v_{i+1}, t^1_{i+1}) is at most $11.452L_{i+1}$. Hence, the length of each edge in P^{i+1}_{i-1} is less or equal than $\max\{11.452L_{i-1}, 11.452L_{i+1}, len(e_{i-1}), len(e_i)\}$. Observe that, by construction, $len(e_{i-1}) \geq cL_{i-1}$, and that $len(e_i) \geq cL_{i+1}$. Hence, as long as $c \geq 11.452$, one between e_{i-1} and e_i is the longest edge in P^{i+1}_{i-1}, and we have only to prove that the distance between w_{i-1} and w_{i+1} is greater than $\max\{len(e_{i-1}), len(e_i)\}$. In the following, refer to Figs. 8.9 and 8.10.

Consider line $l^{4,5}_{i-1}$ separating wedges W^4_{i-1} and W^5_{i-1}, and consider line $l^{1,2}_i$ separating wedges W^1_i and W^2_i. By construction such lines are parallel. Further, T^2_{i-1} is contained in the half-plane delimited by $l^{4,5}_{i-1}$ and not containing $l^{1,2}_i$. Notice that the distance between $l^{4,5}_{i-1}$ and $l^{1,2}_i$ is exactly $len(e_{i-1})$. We claim that, for a suitable constant c, T^1_{i+1} is entirely contained in the half-plane delimited by $l^{1,2}_i$ and not containing $l^{4,5}_{i-1}$. The claim clearly implies that the distance between w_{i-1} and w_{i+1} is greater or equal than $len(e_{i-1})$.

Since the length of edge (v_{i+1}, t^1_{i+1}) is $L_{i+1}/(2 \cdot \tan(2.5)) \leq 11.452L_{i+1}$ and since all the points of Δ^1_{i+1} are at distance at most $L_{i+1}/2$ from t^1_{i+1}, then no point of Δ^1_{i+1} is at distance greater than $11.952L_{i+1}$ from v_{i+1}. Hence, Δ^1_{i+1} is enclosed inside an isosceles triangle $\overline{\Delta}$ having v_{i+1} as a vertex incident to a 5-degree angle, and having two sides (v_{i+1}, v^C_{i+1}) and (v_{i+1}, v^D_{i+1}) of length $11.952L_{i+1}/cos(2.5) \leq 11.964L_{i+1}$ lying on the line $l^{1,2}_{i+1}$ separating wedges W^1_{i+1} and W^2_{i+1}, and on the line $l^{2,3}_{i+1}$ separating wedges W^2_{i+1} and W^3_{i+1}, respectively.

We show that, for a suitable value of c, $\overline{\Delta}$ is entirely contained in the half-plane delimited by $l^{1,2}_i$ and not containing $l^{4,5}_{i-1}$. First, observe that v^C_{i+1} is the point of $\overline{\Delta}$ closer to $l^{1,2}_i$. The distance between v_{i+1} and $l^{1,2}_i$ is equal to $len(e_i) \cdot \sin(20) \geq 0.342cL_{i+1}$. The distance between v^C_{i+1} and v_{i+1} in the direction orthogonal to $l^{1,2}_i$ is $11.964L_{i+1} \cdot \cos(20) \leq 11.243L_{i+1}$. It follows that, as long as $0.342cL_{i+1} \geq 11.243L_{i+1}$, i.e., as long as $c \geq 32.875$, v^C_{i+1} (and hence $\overline{\Delta}$ and Δ^1_{i+1}) is in the half-plane delimited by $l^{1,2}_i$ and not containing $l^{4,5}_{i-1}$.

In analogous way, it can be proved that, as long as $c \geq 32.875$, the distance between w_{i-1} and w_{i+1} is greater than $len(e_i)$. Hence, as long as $c \geq 32.875$,

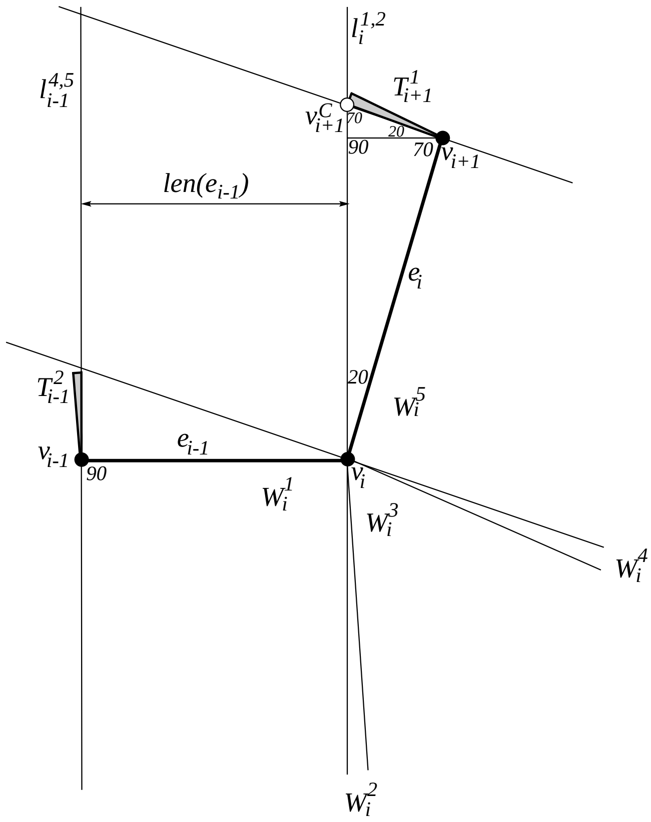


Figure 8.9: Illustration for the proof that the MST condition is satisfied for any node in $T_{i-1}^2 \cup \{v_{i-1}\}$ and any node in $T_{i+1}^1 \cup \{v_{i+1}\}$, for $i = 2, 3, \cdots, k-2$.

the straight-line segment between w_{i-1} and w_{i+1} is longer than every edge in the path P_{i-1}^{i+1} connecting w_{i-1} and w_{i+1} in T, and hence it does not belong to the MST of the points of the drawing.

The length of P. We bound the length of P as a function of the lengths L_i's. As in the binary case, since $len(e_i) = max\{cL_i, cL_{i+1}\}$, and since $len(e_i) \geq 1$, for every $1 \leq i < k$, then $len(e_i) < cL_i + cL_{i+1}$. It follows that $\sum_{i=1}^{k-1} len(e_i) \leq 2c \sum_{i=1}^{k-1} L_i$.

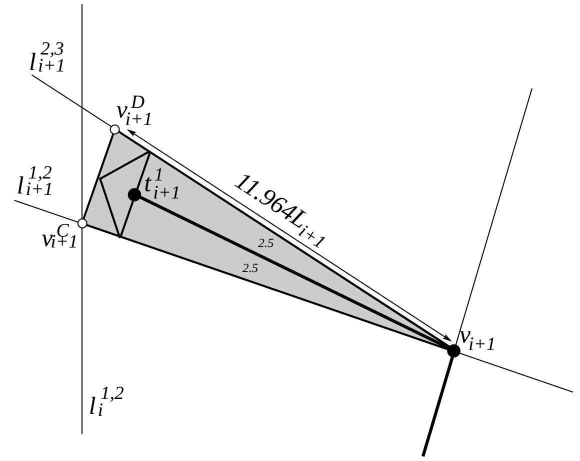


Figure 8.10: Triangle $\overline{\Delta}$, shaded in the picture, containing edge (v_{i+1}, t_{i+1}) and subtree T^1_{i+1}.

The area of the drawing is polynomial. We now compute the length of $h(C)$, i.e., of the hypotenuse of an isosceles right triangle that contains the whole drawing, that has r as midpoint of its hypotenuse, and that has the hypotenuse forming angles of 160, 20, and 180 degrees with edge (v_1, v_2). In the following refer to Fig. 8.11. Notice that the length of the longest edge of the drawing is at most equal to $h(C)$, while the length of the shortest edge of the drawing is at least 1, by construction.

The computation of the area of the drawing proceeds as in the binary case. We first notice that the drawing of P (without the drawing of subtrees T^1_i's and T^2_i's) is contained inside an isosceles triangle Δ_e such that:

- Δ_e has two angles of 20 degrees and one angle of 140 degrees;
- r is the vertex of Δ_e incident to the 140-degree angle;
- one side of Δ_e contains edge (v_1, v_2);
- the distance between r and the side of Δ_e opposite to r is $2c\sum_{i=1}^{k-1} L_i$.

In fact, the length of P is at most $2c\sum_{i=1}^{k-1} L_i$, and, no edge of P forms an angle of less than 20 degrees with a vertical line.

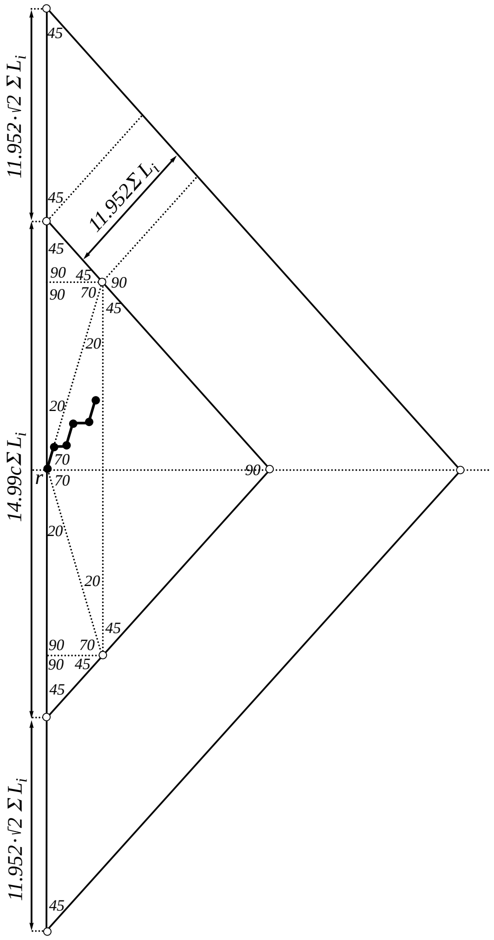


Figure 8.11: Bounding the constructed drawing with an isosceles right triangle.

Consider the smallest isosceles right triangle Δ^* that contains Δ_e completely, that has r as midpoint of its hypotenuse, and that has the hypotenuse forming angles of 160, 20, and 180 degrees with edge (v_1, v_2). Easy trigonometric calculations show that the hypotenuse of Δ^* has length at most $2(1 + 1/\tan(20))(2c\sum_{i=1}^{k-1} L_i) < 14.99c\sum_{i=1}^{k-1} L_i$.

Consider the smallest isosceles right triangle Δ that contains Δ^*, that has r as midpoint of its hypotenuse, that has the hypotenuse forming angles of 160, 20, and 180 degrees with edge (v_1, v_2), and such that every point on one of its catheti has distance at least $11.952 \sum_{i=1}^{k-1} L_i$ from any point of Δ^*. It is easy to see that Δ contains the whole drawing, namely it contains P since it contains Δ^*, and it contains each subtree T_i^1 and T_i^2, since T_i^1 and T_i^2 can stick outside Δ^* by at most $L_i/2 + 11.452L_i = 11.952L_i \leq 11.952 \sum_{i=1}^{k-1} L_i$. Notice that the hypotenuse of Δ has length at most $14.99c \sum_{i=1}^{k-1} L_i + 2(11.952\sqrt{2} \sum_{i=1}^{k-1} L_i)$. By choosing $c = 32.875$, the drawing of T is an MST embedding, and the length of $h(C)$ is bounded by $14.99 \cdot 32.875 \sum_{i=1}^{k-1} L_i + 2(11.952\sqrt{2} \sum_{i=1}^{k-1} L_i) < 526.602 \sum_{i=1}^{k-1} L_i$.

Lemma 8.2 *The length of* $h(C)$ *is at most* $526.602 \sum_{i=1}^{k-1} L_i$.

Let $\alpha = 526.602$. We express $h(C)$ as a function of the number of nodes of the tree. Denote by $h(n)$ the maximum length of $h(C)$ when the input tree has n nodes. It can be inductively proved that $h(n) \leq n^{\log_2(3\alpha)}$. However, this is done by using *exactly* the same arguments and calculations that we used for the binary case, and hence such arguments and calculations are omitted here. Then, we conclude that $h(n) \leq n^{\log_2 1579.805} = O(n^{10.626})$.

Finally, since the area of the drawing is the square of the length of its side, we get the following:

Theorem 8.4 *Every ternary tree with* n *vertices admits an MST drawing in* $O(n^{21.252})$ *area.*

8.6 Conclusions and Open Problems

In this chapter, we have shown algorithms for constructing MST embeddings of trees with maximum degree 4 in polynomial area. It would be interesting to understand how much the bounds achieved by our algorithms can be improved by modifying the constant angles in the geometric constructions we have shown. In the case of complete binary trees, a construction similar to the one we presented for complete ternary trees achieve a slightly better bound than the one claimed in Theorem 8.1 (we still opted for providing the construction of Sect. 8.2, which is particularly simple).

It is an obvious and important open problem to determine whether polynomial-area suffices for constructing MST embeddings of trees with degree 5 or, instead, the conjecture of Monma and Suri is correct [MS92].

Open Problem 8.1 *Which are the asymptotic bounds for the area requirements of MST embeddings of trees with maximum degree 5?*

We remark that an exponential area lower bound can be quite easily obtained if the order of the edges incident to the nodes of the tree is fixed. In order to prove such a lower bound, it is sufficient to consider a 5-regular caterpillar, i.e. a caterpillar T such that all its non-leaf nodes have degree 5, that has the following property: Removing all the leaves turns T into a path $(v_1, v_2, \cdots, v_k)$, in which edge (v_i, v_{i+1}) is the last edge incident to v_i, in the clockwise order of the edges incident to v_i starting at (v_{i-1}, v_i), for each $i = 2, 3, \ldots, k-1$ (see Fig. 8.12).

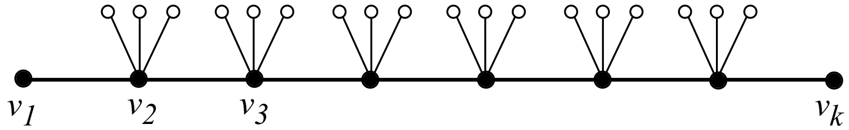


Figure 8.12: A 5-regular caterpillar in which the path always turns right.

However, we expect that there exist degree-5 trees for which exponential area is required in any MST embedding, even if the order of the edges incident to each node is not fixed. In fact, we have the following conjecture:

Conjecture 8.1 *The* 5*-regular caterpillar requires exponential area.*

Part IV

Clustered Graphs

Chapter 9

Straight-line, Poly-line, Orthogonal, and Upward Drawings of Clustered Trees

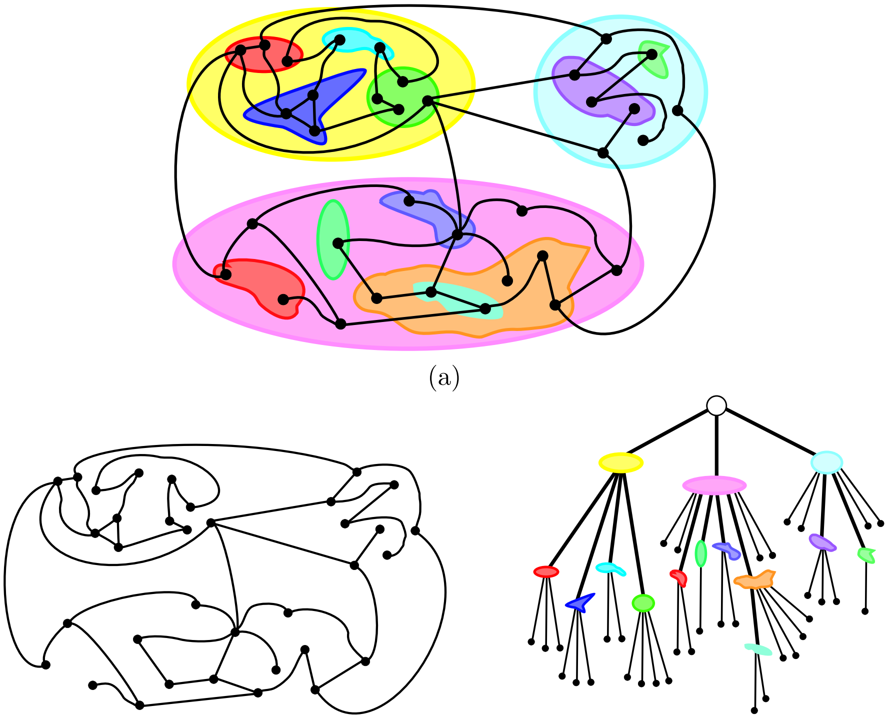
In this chapter[1] we consider straight-line, poly-line, orthogonal, and upward drawings of directed trees in small area. For each of the drawing conventions, we consider models in which the clusters are drawn as rectangles, as convex polygons, and as non-convex polygons. We show many results that put in evidence how drawing clustered trees has many sharp differences with respect to drawing "plain" trees. In fact, we show that there exist clustered trees that do not have any drawing in certain standards and clustered trees that require exponential area in other standards. On the contrary, for many drawing conventions we show efficient algorithms that allow to draw clustered trees with polynomial asymptotically optimal area.

9.1 Introduction

A *clustered graph* is a pair $C = (G, T)$, where G is a graph and T is a rooted tree such that the leaves of T are the vertices of G. Graph G and tree T are called *underlying graph* and *inclusion tree*, respectively. Fig. 9.1 shows a clustered graph. A *clustered tree* is a clustered graph whose underlying graph is a tree. The clustered graph in Fig. 9.2 is a clustered tree. Each internal

[1] The contents of this chapter are a joint work with Giuseppe Di Battista and Guido Drovandi, appeared in [DDF07] and to appear in [DDF09].

node ν of T corresponds to the subset $V(\nu)$ of the vertices of G (called *cluster*) that are the leaves of the subtree rooted at ν. The subgraph of G induced by $V(\nu)$ is denoted by $G(\nu)$, where ν is a cluster of T. If each cluster induces a connected subgraph of G, then C is *c-connected*. The clustered tree in Fig. 9.2 is not c-connected.



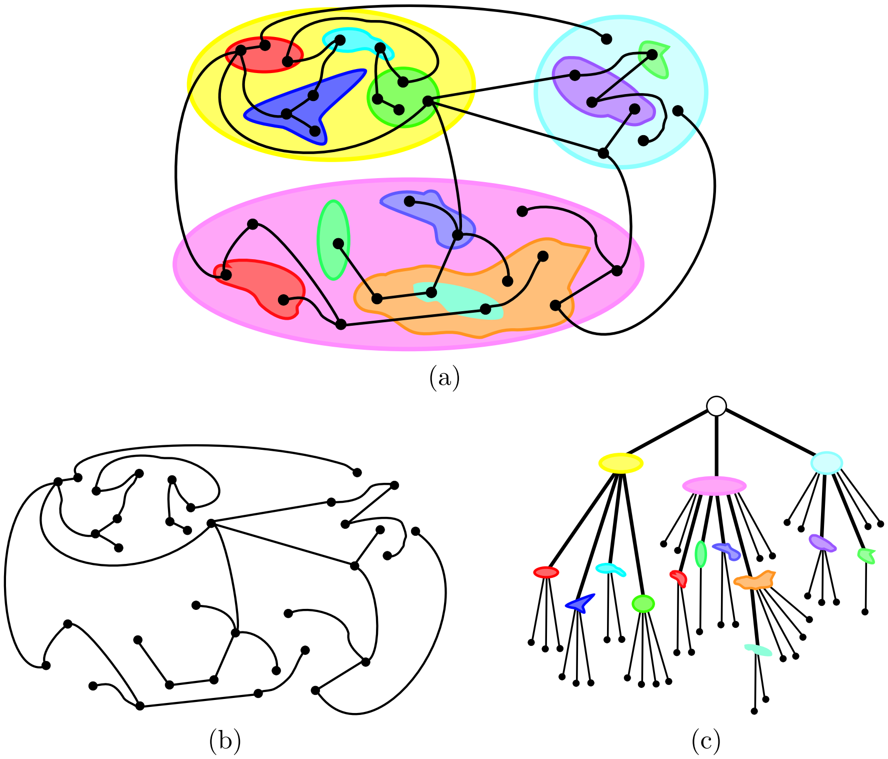
(a)

(b)

(c)

Figure 9.1: (a) A c-planar drawing of a clustered graph $C = (G, T)$. (b) The underlying graph G of C. (c) The inclusion tree T of C.

Clustered graphs are widely used in applications where it is needed at the same time to show relationships between entities and to group entities with semantic affinities. For example, in a social network representing the working relationships between the employees of a company it might be desirable to group into clusters the people of each department.


(a)

(b)

(c)

Figure 9.2: (a) A c-planar drawing of a clustered tree $C = (G, T)$. (b) The underlying graph G of C. Note that G is a tree. (c) The inclusion tree T of C.

Visualizing clustered graphs turns out to be a difficult problem, due to the simultaneous need for a readable drawing of the underlying structure and for a good rendering of the recursive clustering relationship. As for the visualization of graphs, the most important aesthetic criterion for a drawing of a clustered graph to be "nice" is commonly regarded to be *planarity*. However, the classical concept of planarity needs a refinement in the context of clustered graphs, in order to deal also with the clustering structure.

A drawing of a clustered graph $C = (G, T)$ consists of a drawing of G and of a representation of each node ν of T as a simple closed region $R(\nu)$ such that: (i) $R(\nu)$ contains the drawing of $G(\nu)$; (ii) $R(\nu)$ contains a region $R(\mu)$

iff μ is a descendant of ν in T; and (iii) the borders of any two regions do not intersect. Consider an edge e and a node ν of T. If e crosses the boundary of $R(\nu)$ more than once, we say that edge e and region $R(\nu)$ have an *edge-region crossing*. A drawing of a clustered graph is *c-planar* if it does not have edge crossings or edge-region crossings and a graph is c-planar if it has a c-planar drawing.

A number of papers have been presented for constructing c-planar drawings of clustered graphs within many drawing conventions, and the problem of determining the area requirements of c-planar drawings of clustered graphs has been considered. Namely, Eades, Feng, and Nagamochi in [EFN99] showed an algorithm to construct $O(n^2)$ area c-planar orthogonal drawings of clustered graphs with maximum degree 4, in which each cluster is represented by an axis-parallel rectangle. Eades *et al.* [EFLN06] proved that every c-planar clustered graph admits a c-planar straight-line drawing, in which all the clusters are drawn as convex regions. In fact, they provided an algorithm for constructing such drawings in exponential area. However, in [FCE95a] it is shown that such a bound is asymptotically optimal in the worst case. All the cited results are more extensively discussed throughout the chapter.

In this chapter we look for algorithms to construct c-planar drawings of clustered trees with efficient area. A clustered tree is a clustered graph such that its underlying graph is a tree. We consider the most investigated drawing standards for the underlying tree (see, e.g., [CGKT02, Cha02, GGT96, GR04]), namely straight-line, poly-line, orthogonal, and upward drawings, and combinations of them. We deal both with c-connected and non-c-connected clustered trees. Further, we consider drawings in which the clusters are represented by rectangles (R-drawings), by convex polygons (C-drawings), and also by eventually non-convex polygons (NC-drawings).

We provide the following results: (i) there exist n-node c-connected clustered trees requiring $\Omega(n^2)$ area in any NC-drawing (and hence in any R- and C-drawing); (ii) every n-node c-connected clustered tree $C = (G, T)$ has a $\Theta(n^2)$ area strictly-upward order-preserving poly-line (strictly-upward non-order-preserving straight-line) R-drawing (and hence C- and NC-drawing); (iii) every n-node c-connected clustered binary tree $C = (G, T)$ has a $\Theta(n^2)$ area strictly-upward order-preserving straight-line (upward order-preserving orthogonal) R-drawing (and hence C- and NC-drawing); (iv) there exist n-node c-connected clustered binary trees that do not admit any straight-line orthogonal R-drawing; (v) there exist n-node non-c-connected clustered binary trees that do not admit any upward C-drawing (and hence any upward R-drawing); (vi) there exist n-node non-c-connected clustered trees requiring exponential area in any straight-line C-drawing (and hence in any straight-line R-drawing); (vii) every n-node non-c-connected clustered tree $C = (G, T)$ has a $\Theta(n^2)$ area order-preserving poly-line R-drawing (and hence C- and NC-drawing); (viii) every n-node c-connected clustered tree $C = (G, T)$ has a $O(n^4)$ area strictly-upward order-preserving straight-line NC-drawing; (ix) every n-node c-connected clustered tree $C = (G, T)$ has a $O(n^3 \log n)$ area straight-line orthogonal NC-drawing.

Tables 9.1 and 9.2 summarize the area requirements of R-drawings and C-drawings of c-connected and non c-connected clustered trees. In the tables, "UB" and "LB" stand for *Upper Bound* and *Lower Bound*, respectively. "Upward" means *upward* when referring to orthogonal drawings and means *strictly upward* otherwise. If the straight-line column does not have a "✓", then the drawing is poly-line. Orthogonal drawings are referring to binary trees. An "X" means that in general a drawing with the corresponding features does not exist.

				R-Drawings				C-Drawings			
upward	straight-line	ordered	orthogonal	UB	ref.	LB	ref.	UB	ref.	LB	ref.
✓	✓			$O(n^2)$	Th. 9.2	$\Omega(n^2)$	Le. 9.1	$O(n^2)$	Th. 9.2	$\Omega(n^2)$	Le. 9.1
✓	✓	✓		**?**	-	$\Omega(n^2)$	Le. 9.1	**?**	-	$\Omega(n^2)$	Le. 9.1
✓		✓		$O(n^2)$	Th. 9.1	$\Omega(n^2)$	Le. 9.1	$O(n^2)$	Th. 9.1	$\Omega(n^2)$	Le. 9.1
✓		✓	✓	$O(n^2)$	Th. 9.4	$\Omega(n^2)$	Le. 9.1	$O(n^2)$	Th. 9.4	$\Omega(n^2)$	Le. 9.1
	✓		✓	**X**		Th. 9.5		**?**	-	$\Omega(n^2)$	Le. 9.1
		✓	✓	$O(n^2)$	[EFN99]	$\Omega(n^2)$	Le. 9.1	$O(n^2)$	[EFN99]	$\Omega(n^2)$	Le. 9.1

Table 9.1: Summary of the results on minimum area drawings of c-connected clustered trees.

The rest of the chapter is organized as follows. In Sect. 9.2 we establish precise definitions about R-drawings, C-drawings, and NC-drawings, and we prove a simple lower bound valid for any considered drawing standard; in Sect. 9.3, we deal with R-drawings and C-drawings of c-connected clustered trees; in Sect. 9.4 we deal with R-drawings and C-drawings non-c-connected clustered trees; in Sect. 9.5 we deal with NC-drawings of c-connected and non-c-connected clustered trees; finally, in Sect. 9.6 we conclude and present some open problems.

				R-Drawings				*C*-Drawings			
upward	straight-line	ordered	orthogonal	UB	ref.	LB	ref.	UB	ref.	LB	ref.
✓				**X**		Th. 9.6		**X**		Th. 9.6	
	✓			**?**	**-**	$\Omega(2^n)$	Th. 9.7	$O(2^n)$	[EFLN06]	$\Omega(2^n)$	Th. 9.7
		✓	✓	$O(n^2)$	[EFN99]	$\Omega(n^2)$	Le. 9.1	$O(n^2)$	[EFN99]	$\Omega(n^2)$	Le. 9.1
		✓		$O(n^2)$	Th. 9.8	$\Omega(n^2)$	Le. 9.1	$O(n^2)$	Th. 9.8	$\Omega(n^2)$	Le. 9.1

Table 9.2: Summary of the results on minimum area drawings of non-*c*-connected clustered trees.

9.2 Preliminaries on R-Drawings, C-Drawings, and NC-Drawings

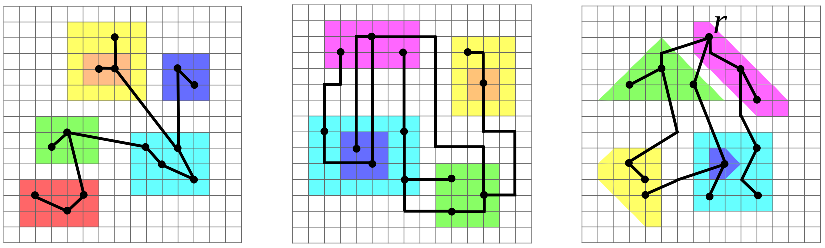
We define the following drawing conventions for clustered graphs. A polygon with vertices having integer coordinates is a *lattice polygon*.

Definition 9.1 *A drawing of a clustered tree $C = (G, T)$ is an NC-drawing (for* Non-Convex*-drawing) if it is c-planar, the vertices of G and the bends on the edges of G (if any) have integer coordinates, and the border of each cluster is a lattice polygon.*

Definition 9.2 *A drawing of a clustered tree $C = (G, T)$ is a C-drawing (for* Convex*-drawing) if it is c-planar, the vertices of G and the bends on the edges of G (if any) have integer coordinates, and the border of each cluster is a convex lattice polygon.*

Definition 9.3 *A drawing of a clustered tree $C = (G, T)$ is an R-drawing (for* Rectangle*-drawing) if it is c-planar, the vertices of G and the bends on the edges of G (if any) have integer coordinates, and the border of each cluster is an axis-parallel rectangle with corners having integer coordinates.*

Notice that by definition an R-drawing is also a C-drawing and a C-drawing is also an NC-drawing. Hence, an area upper bound obtained for R-drawings is also an upper bound for C-drawings and for NC-drawings. On the contrary, a lower bound for NC-drawings implies a lower bound for C-drawings and for R-drawings. Figure 9.3 shows examples of R-drawings and C-drawings within different drawing standards for the underlying tree.



(a) (b) (c)

Figure 9.3: (a) A straight-line R-drawing. (b) An orthogonal R-drawing. (c) A poly-line upward C-drawing.

The following lemma is easy to prove. Throughout the rest of the chapter, given a clustered tree (G, T), let $G(v)$ $(T(v))$ denote the subtree of G (resp. of T) rooted at v.

Lemma 9.1 *There exist n-vertex clustered trees requiring $\Omega(n^2)$ area in any NC-drawing.*

Proof: Consider a clustered tree $C = (G, T)$ such that T has height h. We show by induction on h that C requires $2h$ width in any NC-drawing. If $h = 1$ then the border of the only cluster of T must be drawn as a simple lattice polygon and hence it must intersect at least two vertical grid lines of the plane. Suppose by induction hypothesis that $2h - 2$ is the minimum width of an NC-drawing of a clustered tree $C' = (G', T')$ such that the height of T' is $h - 1$. Consider a clustered tree $C = (G, T)$ such that the height of T is h. Clearly, there exists a subtree $T(\nu)$ of T rooted at a child ν of the root r of T that has height $h - 1$. By induction hypothesis every NC-drawing Γ' of $C' = (G(\nu), T(\nu))$ requires $2h - 2$ width. Draw the polygon P representing r. By definition of NC-drawing, P surrounds Γ' and its vertices have integer coordinates, so P must touch the vertical grid line one unit to the left of the leftmost vertical line intersecting Γ' and must touch the vertical grid line one unit to the right of the rightmost vertical line intersecting Γ'. It follows that C requires the width of Γ' plus two units, so it requires $2h$ width. Analogously, the minimum height of an NC-drawing of a clustered tree with height h is $2h$. Since there exist clustered trees $C = (G, T)$ such that T has height $\Omega(n)$, the lemma follows. □

9.3 R-Drawings and C-Drawings of C-Connected C-Trees

In this section we consider R-drawings and C-drawings of c-connected c-trees.

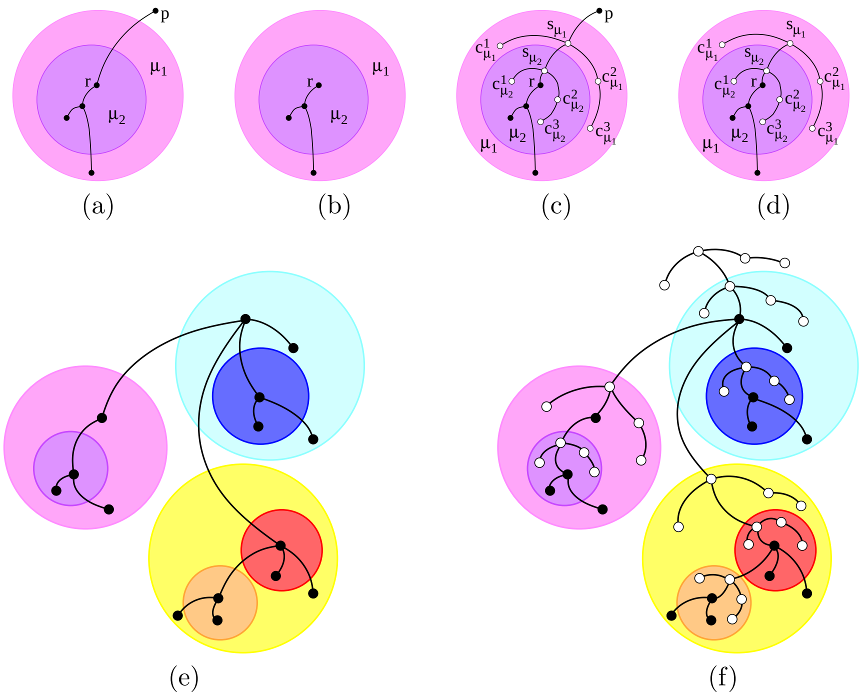
We show that quadratic area is sufficient (and necessary) to construct R-drawings of c-connected clustered trees in which the underlying tree is represented within several drawing standards. Namely, we present an algorithm for constructing $\Theta(n^2)$ area strictly upward order-preserving poly-line R-drawings of n-node c-connected clustered trees. Then, we show how to slightly modify such an algorithm to obtain different kinds of drawings.

Let $C = (G, T)$ be a c-connected clustered tree. We first perform an augmentation of C. Namely, for each cluster of T we add dummy vertices and edges to G as follows. Consider each node r of G, if r is the root of k subtrees induced by k clusters, i.e. $\Sigma_r = \{\mu_1, \dots, \mu_k\}$ (suppose μ_i is the parent of μ_{i+1} in T for each $1 \leq i \leq k-1$), add a path composed by nodes $s_{\mu_1}, s_{\mu_2}, \dots, s_{\mu_k}$ (a node for each cluster belonging to Σ_r) to G as follows: If the parent p of r exists (see Fig. 9.4 (a)) then split the edge (p, r) into edges (p, s_{μ_1}) and (s_{μ_k}, r) (Fig. 9.4 (c)); otherwise (Fig. 9.4 (b)) add only the edge (s_{μ_k}, r) (Fig. 9.4 (d)). In any case at each dummy vertex s_μ add two dummy children c^1_μ and c^2_μ and a dummy node c^3_μ as child of c^2_μ; these four dummy vertices belong to cluster μ. The counter-clockwise order of the children of s_{μ_i} ($\mu_i \in \Sigma_r$) is: If $1 \leq i < k$, $c^1_{\mu_i}$, $s_{\mu_{i+1}}$, and $c^2_{\mu_i}$; otherwise $c^1_{\mu_k}$, r, and $c^2_{\mu_k}$. After having performed the described augmentation on each cluster, we obtain a clustered tree $C' = (G', T')$. We call r' the root of G'. Figure 9.4 (e) shows a clustered tree C and Figure 9.4 (f) the augmentation of C. A pseudo-code description of the above phase is given in Fig. 9.5.

We now construct a strictly upward drawing of G'. From now on, $x(v)$ and $y(v)$ denote the coordinates assigned to a node v. Further, denote by $p(v)$ the parent of a node v. First, assign an x-coordinate to each node in G' by means of a depth-first traversal of G'. Set $x(r') = 0$. Then, suppose that the x-coordinate has already been assigned to a node v. Let $v_1, v_2, \dots, v_m$ be the children of v in counter-clockwise order. Set $x(v_1) = x(v)$; for each child v_i of v, $i = 2, \dots, m$, set $x(v_i) = 1 + \max_{u \in G'(v_{i-1})}\{x(u)\}$ (see Fig. 9.6). A pseudo-code description of this step is given in Fig. 9.7.

Concerning the y-coordinates we adopt the following general strategy. We perform a traversal of G' based on the clustering hierarchy of C'. Such a traversal has the following properties:

- We maintain a *current cluster* such that, when node $\mu \in T'$ is the current



(a) (b) (c) (d)

(e) (f)

Figure 9.4: (a) The root of $G(\mu_1)$ is not the root of the underlying tree G. (b) The root of $G(\mu_1)$ is the root of the underlying tree G. (c)–(d) $G(\mu_1)$ augmented with dummy vertices and edges. (e) A clustered tree $C = (G, T)$. (f) Clustered tree $C' = (G', T')$ obtained by the augmentation of C with dummy vertices and edges.

cluster we perform a depth-first traversal of $G'(\mu)$.

- When a node v is visited, the current cluster is set equal to the smallest cluster containing v and containing at least one not yet visited node.

More in detail, the y-coordinates assignment is as follows (see Fig. 9.9): Set $y(r') = 0$. Let μ_r be the smallest cluster containing r'. Set the current cluster to be μ_r. Now suppose that node $\mu \in T'$ is the current cluster.

- If there is more than one node in $V'(\mu)$ that is not yet visited, then consider the first not yet visited node $v \in V'(\mu)$ that is encountered in a

```
**Augmentation Algorithm for a C-Connected Clustered Tree**
*input:* A $c$-connected clustered tree $C(G,T)$
  **for all** cluster $\mu \in T$ **do**
    $v = root(\mu)$
    $s_\mu$, $c_\mu^1$, $c_\mu^2$, $c_\mu^3$ = new vertices
    add vertices $s_\mu$, $c_\mu^1$, $c_\mu^2$ and $c_\mu^3$ to cluster $\mu$
    $root(\mu) = s_\mu$
    $last(\mu) = c_\mu^3$
    **if** $parent(v)$ is not NIL **then**
      substitute $v$ with $s_\mu$ in $children(parent(v))$
    **end if**
    $parent(s_\mu) = parent(v)$
    $parent(c_\mu^1) = parent(v) = parent(c_\mu^2) = s_\mu$
    $parent(c_\mu^3) = c_\mu^2$
    $children(s_\mu) = \{c_\mu^1, v, c_\mu^2\}$ in counter-clockwise order
    $children(c_\mu^2) = \{c_\mu^3\}$
  **end for**
```

Figure 9.5: The augmentation algorithm.

depth-first traversal of $G'(\mu)$.

  - If the smallest cluster ν containing v is the same cluster or is a descendant of the smallest cluster containing $p(v)$, then set $y(v) = y(p(v)) - 1$; cluster ν is the new current cluster.
  - Otherwise (the smallest cluster ν containing v is not the same cluster or a descendant of the smallest cluster containing $p(v)$) set $y(v)$ equal to the minimum y-coordinate of a node in the biggest cluster containing $p(v)$ and not containing v minus one; cluster ν is the new current cluster.

- If there is exactly one node in $V'(\mu)$ that is not yet visited, then such a node is c_μ^3; set $y(c_\mu^3)$ equal to the minimum y-coordinate of a node in $V'(\mu)$ minus one; set the current cluster to be μ.

- If all nodes in $V'(\mu)$ are already visited, then set the current cluster to be the parent of μ.

From the drawing of G', obtained with the above technique, we construct a strictly upward order-preserving poly-line R-drawing as follows (Fig. 9.10). For

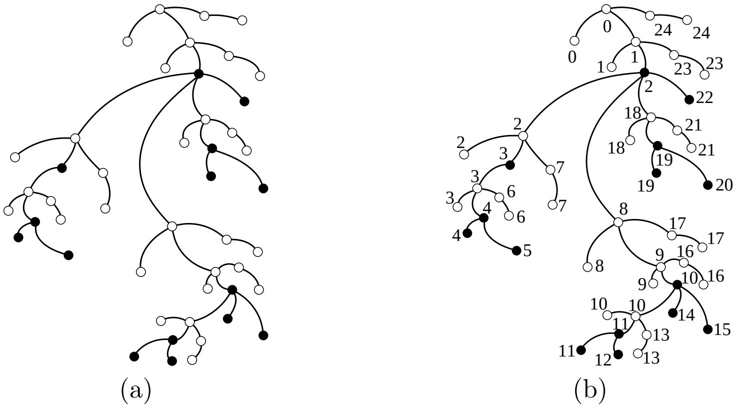


Figure 9.6: (a) The underlying tree G' of the clustered tree C' of Fig. 9.4 (f). (b) The x-coordinates assignment to the vertices of G'.

```
X-Coordinates Assignment for a C-Connected Clustered Tree
input: The root of an augmented c-connected clustered tree $C(G,T)$
  Initialization
  $leaf\_counter = 0$
  Procedure AssignXCoord($r$)
  if $children(r) = \emptyset$ then
    $x(r) = leaf\_counter$
    $leaf\_counter = leaf\_counter + 1$
  else
    for all $v \in children(r)$ in counter-clockwise order do
      AssignXCoord($v$)
    end for
    $v$ = the first child of $r$
    $x(r) = x(v)$
  end if
```

Figure 9.7: The x-coordinates assignment.

each cluster μ remove vertices s_μ, c_μ^1, c_μ^2, and c_μ^3 and their incident edges, and insert a rectangle R_μ:$[x(c_\mu^1), x(c_\mu^3)] \times [y(c_\mu^3), y(s_\mu)]$ (see Fig. 9.11) representing μ in the final drawing. Draw the edges of G: For each edge $(p(v), v)$ in G, if $y(v) = y(p(v)) - 1$ then draw a straight-line segment between $p(v)$ and v,

```
$Y$-Coordinates Assignment for a C-Connected Clustered Tree
input: The root of an augmented $c$-connected clustered tree $C(G,T)$, the
    root of $T$ and 0 (the $y$-coordinate of the upmost vertex)
output: The $y$-coordinate of the lower vertex
  Initialization
  for all $\mu \in T$ do
    $minY(\mu) = 0$
  end for
  Procedure AssignYCoord($v$, $\mu$, $yCoord$)
  if $y(v)$ is not set then
    if $v = last(\mu)$ then
      $y(v) = minY(\mu) - 1$
      remove cluster $\mu$
      return $y(v) - 1$
    else
      $y(v) = yCoord$
      $yCoord = y(v) - 1$
    end if
  end if
  $newYCoord = yCoord$
  for all $\nu \in clusters(v) \leq \mu$ do
    $minY(\nu) = \min\{minY(\nu), y(v)\}$
    for all $s \in children(v)$ in counter-clockwise order do
      if $clusters(s) \cap \{\nu\} \neq \emptyset$ then
        $newYCoord = \text{AssignYCoord}(s, \nu, yCoord)$
      end if
    end for
    $yCoord = newYCoord$
  end for
  return $yCoord$
```

Figure 9.8: The y-coordinates assignment.

otherwise ($y(v) < y(p(v)) - 1$) draw a polygonal line composed of two segments, the first between $p(v)$ and point $(x(v), y(p(v)) - 1)$, and the second between point $(x(v), y(p(v)) - 1)$ and v. Figure 9.12 (b) shows a drawing constructed by the described algorithm, Figure 9.12 (a) shows the same drawing before the removal of the dummy vertices. We have:

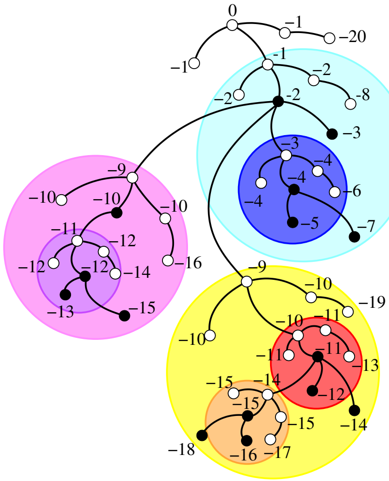


Figure 9.9: The y-coordinates assignment to the vertices of the underlying tree G' of the clustered tree C' of Fig. 9.4 (f).

Theorem 9.1 *For every n-node c-connected clustered tree $C = (G, T)$ a $\Theta(n^2)$ area strictly upward order-preserving poly-line R-drawing can be constructed in $O(n^2)$ time.*

Proof: The drawing Γ obtained by applying the previously described algorithm is strictly upward, order-preserving, and poly-line by construction. All the vertices, bends, and corners of the rectangles representing clusters have integer coordinates. Further, each cluster μ contains all the vertices and clusters belonging to $T(\mu)$. This is because the coordinates of the box $R(\mu)$ are obtained from the coordinates of the dummy vertices s_μ and c^3_μ: Since such vertices are respectively the first and the last vertex considered in the coordinates' assignment for the vertices in $V'(\mu)$, then they have respectively the smallest x- and y-coordinate and the greatest x- and y-coordinate among all real and dummy vertices in $V'(\mu)$. Hence, the rectangle delimited by such coordinates contains all the vertices and clusters belonging to $T(\mu)$.

The planarity of the drawing of G is proved as follows. Consider any pair of edges $e_1 = (u_1, v_1)$ and $e_2 = (u_2, v_2)$ such that u_1 and u_2 are parents of v_1 and v_2, respectively. Assume, w.l.o.g., that if u_1 and u_2 are on the same path from the root to a leaf then u_1 is an ancestor of u_2. We distinguish three cases:

```
Drawing Algorithm for a C-Connected Clustered Tree
input: The root of a $c$-connected clustered tree $C(G,T)$
  Procedure RDrawing($r$)
  if $r$ is a dummy vertex then
     $v$ = the second child of $r$
     $c_1$ = the first child of $r$
     $c_2$ = the last (the third) child of $r$
     $c_3$ = the only child of $c_2$
     draw a horizontal line between $(x(c_1), y(r))$ and $(x(c_3), y(r))$
     draw a horizontal line between $(x(c_1), y(c_3))$ and $(x(c_3), y(s_3))$
     draw a vertical line between $(x(c_1), y(r))$ and $(x(r), y(c_3))$
     draw a vertical line between $(x(c_3), y(r))$ and $(x(c_3), y(c_3))$
     $parent(v) = parent(r)$
     remove vertices $r$, $c_1$, $c_2$, $c_3$
     RDrawing($v$)
  else
     draw a vertex in $(x(r), y(r))$
     if $parent(r)$ is not NIL then
        $p = parent(r)$
        case polyline drawing:
            draw a straight-line between $(x(p), y(p))$ and $(x(r), y(p)+1)$
            draw a vertical line between $(x(r), y(p)+1)$ and $(x(r), y(r))$
        case orthogonal drawing:
            draw a horizontal line between $(x(p), y(p))$ and $(x(r), y(p))$
            draw a vertical line between $(x(r), y(p))$ and $(x(r), y(r))$
        case straight-line drawing:
            draw a straight-line between $(x(p), y(p))$ and $(x(r), y(r))$
     end if
     for all $v \in children(r)$ do
        RDrawing($v$)
     end for
  end if
```

Figure 9.10: The R-drawing algorithm.

- If v_1 is an ancestor of u_2 or coincides with u_2, then u_1 is an ancestor of u_2 and edges e_1 and e_2 do not cross by the upwardness of Γ.

- If u_1 and u_2 coincide, consider the at most two segments s_1^a and s_1^b (resp. s_2^a and s_2^b) representing edge e_1 (resp. edge e_2) in Γ, where s_1^a connects

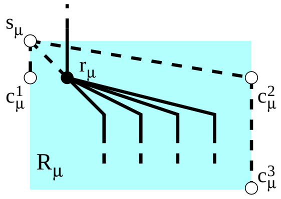


Figure 9.11: The rectangle R_μ defined by the four dummy vertices associated to cluster μ. Dashed lines represent the edges in the augmented cluster tree C'.

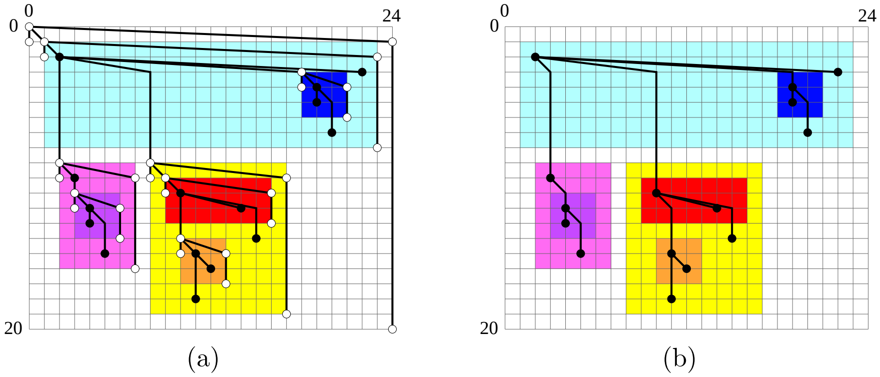


Figure 9.12: (a) The drawing of the clustered tree C of Fig. 9.4 (e) constructed by the algorithm described in Sect. 9.3 before the removal of the dummy vertices. (b) The drawing of C after the removal.

u_1 with $(x(v_1), y(u_1) - 1)$ and s_1^b connects $(x(v_1), y(u_1) - 1)$ with v_1 (resp. where s_2^a connects $u_1 = u_2$ with $(x(v_2), y(u_1) - 1)$ and s_2^b connects $(x(v_2), y(u_1) - 1)$ with v_2). Segment s_1^a cannot cross s_2^b and segment s_2^a cannot cross s_1^b, since they lie in disjoint y-intervals; segment s_1^a cannot cross s_2^a since they are incident to the same vertex and have different slopes; segment s_1^b cannot cross s_2^b since they lie on different vertical lines.

- If u_1 is an ancestor of u_2 but v_1 is not, then consider the at most two segments s_1^a and s_1^b representing edge e_1 in Γ, where s_1^a connects u_1 with $(x(v_1), y(u_1) - 1)$ and s_1^b connects $(x(v_1), y(u_1) - 1)$ with v_1. Segment s_1^a

can not cross e_2 since they lie in disjoint y-intervals. Segment s_1^b can not cross e_2 since they lie in disjoint x-intervals.

- If u_1 is not ancestor of u_2 and if they are not the same node, then they are not on the same path from the root to a leaf. Consider the lowest common ancestor lca of u_1 and u_2. Edges e_1 and e_2 belong to different subtrees among the subtrees rooted at the children of lca and hence they lie in in disjoint x-intervals.

If μ is the parent of ν, we have that $R(\nu)$ is contained in $R(\mu)$: Namely, all the vertices belonging to $V(\nu)$ have an x-coordinate that is strictly greater than the x-coordinate of the root s_μ of $G(\mu)$ and that is strictly lesser than the x-coordinate of c_μ^3 (the rightmost vertex of $G(\mu)$). Also, all the vertices belonging to $V(\nu)$ have a y-coordinate that is strictly lesser than the y-coordinate of the root s_μ of $G(\mu)$ and that is strictly greater than the y-coordinate of c_μ^3 (the bottommost vertex of $G(\mu)$).

Region-region crossings do not occur: Namely, consider two distinct clusters μ and ν, none of which is an ancestor of the other in T. If the roots $r(\mu)$ and $r(\nu)$ of $G(\mu)$ and $G(\nu)$, respectively, are such that $r(\mu)$ is an ancestor of $r(\nu)$ in G or viceversa, then the regions representing μ and ν in Γ lie in disjoint y-intervals. Otherwise ($r(\mu)$ and $r(\nu)$ are not on the same path from the root to a leaf in G), the regions representing μ and ν in Γ lie in disjoint x-intervals.

Concerning edge-region crossings, if the y-interval of an edge $e = (u, v)$ has intersection with the y-extension of a cluster μ three cases are possible:

- Edge e belongs to $G(\mu)$. In this case e is internal to the region $R(\mu)$ representing μ in Γ, since by construction the drawing of e in Γ is contained inside the smallest rectangle $R(e)$ with sides parallel to the axis that contains both u and v. Clearly $R(e)$ is internal to $R(\mu)$.

- Edge e does not belong to $G(\mu)$, but it is incident to a vertex of $G(\mu)$. Since $R(\mu)$ is a rectangle with sides parallel to the axes, since the second segment of e, if any, is vertical, then e crosses the border of μ exactly once.

- If edge e does not belong to $G(\mu)$ and is not incident to a vertex of $G(\mu)$, then there exists a node $\overline{r}$ in G such that $G(\mu)$ and e belong to two non overlapping subtrees of $G(\overline{r})$. This implies that e and $R(\mu)$ lie in disjoint x-intervals.

Concerning the area requirement, observe that for every horizontal or vertical line intersecting Γ there is at least one node of G or one side of a rectangle representing a cluster. Since there are n vertices and $O(n)$ sides, both the height and the width of Γ are at most linear, so the area upper bound follows. By Lemma 9.1, quadratic area is necessary in the worst case. Concerning the running time, it's easy to see that the algorithm can be implemented to run in $O(n^2)$ time. □

The above described algorithm can be slightly modified in order to produce R-drawings within different drawing conventions for the underlying tree.

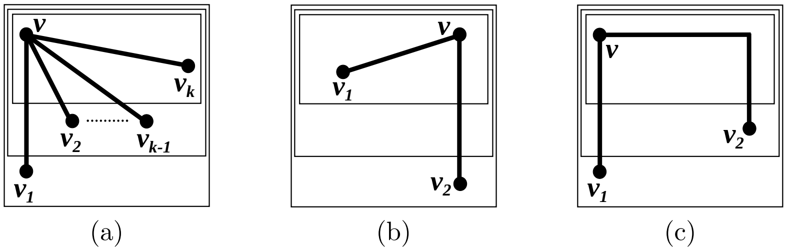


Figure 9.13: (a) Strictly upward non-order-preserving straight-line R-drawings of c-connected clustered trees. (b) Strictly upward order-preserving straight-line R-drawings of c-connected binary clustered trees. (c) Upward orthogonal order-preserving straight-line R-drawings of c-connected binary clustered trees.

Theorem 9.2 *For every n-node c-connected clustered tree $C = (G, T)$ a $\Theta(n^2)$ area strictly upward non-order-preserving straight-line R-drawing can be constructed in $O(n^2)$ time.*

Proof: First, counter-clockwise order the children of each node v so that a child v_i of v coming before a child v_{i+1} of v is such that the smallest cluster containing both v_i and v is an ancestor or is the same cluster of the smallest cluster containing both v_{i+1} and v. Second, augment $C = (G, T)$ to a clustered tree $C' = (G', T')$ and assign x- and y-coordinates to the nodes of G' as in the previous algorithm. The x- and y-coordinates' assignment of the algorithm and the order of the children of each node ensure that $x(v_{i+1}) > x(v_i)$ and $y(v_{i+1}) \geq y(v_i)$, for each pair of consecutive nodes v_i and v_{i+1} of a node v (see Fig. 9.13 (a)). Hence, after replacing dummy vertices with rectangles representing clusters, the edges connecting each node to its children can be drawn as straight-line segments, without introducing crossings. □

Theorem 9.3 *For every n-node c-connected binary clustered tree $C = (G, T)$ a $\Theta(n^2)$ area strictly upward order-preserving straight-line R-drawing can be constructed in $O(n^2)$ time.*

Proof: In this case the previously described algorithm loses the invariant that the root v of a subtree $G(v)$ is the leftmost node in $G(v)$. Hence, the x-coordinates assignment changes slightly, while the y-coordinates assignment remains the same. Compute the y-coordinate of each node as in the previous algorithm. For each non-leaf node $v \in G$, let v_1 and v_2 be the children of v and let μ_1 (μ_2) be the smallest cluster containing both v and v_1 (containing both v and v_2). If μ_1 is an ancestor of or is the same cluster as μ_2, set the x-coordinate of v to $x(v_1)$. Otherwise (if μ_2 is an ancestor of μ_1), set the x-coordinate of v to $x(v_2)$ (see Fig. 9.13 (b)). It's easy to see that the edges of G can be drawn as straight-line segments without introducing crossings. □

Theorem 9.4 *For every n-node c-connected binary clustered tree $C = (G, T)$ a $\Theta(n^2)$ area upward orthogonal order-preserving R-drawing can be constructed in $O(n^2)$ time.*

Proof: What changes in this case, with respect to the original formulation of the algorithm, is just the position of the bend on every edge between a node v and its second child v_2, in the counter-clockwise order of the children of v. Namely, such an edge must be orthogonal, hence it goes horizontally till reaching $x(v_2)$, and then goes down to v_2 vertically (see Fig. 9.13 (c)). □

Contrasting with the above positive results, we prove the following theorem, that also contrasts with the fact that each binary tree has an orthogonal straight-line drawing [DETT99].

Theorem 9.5 *There exists a c-connected c-planar binary clustered tree that does not admit any orthogonal straight-line R-drawing.*

Proof: Consider the clustered tree $C = (G, T)$ defined as follows: G is a complete rooted binary tree with 31 vertices; all the non-leaf vertices of G belong to the same cluster α that is the only non-root cluster, and all the leaves of G do not belong to α. It's easy to see that C is c-planar. Fig. 9.14 (a) shows a c-planar drawing of C.

Consider any orthogonal straight-line R-drawing Γ of C. Consider the rectangle A representing α in Γ. Let r be the root of G, let $u_1, \dots, u_8$ be the 8 vertices that are leaves in $G(\alpha)$, and let $v_1, \dots, v_4$ be the corners of A.

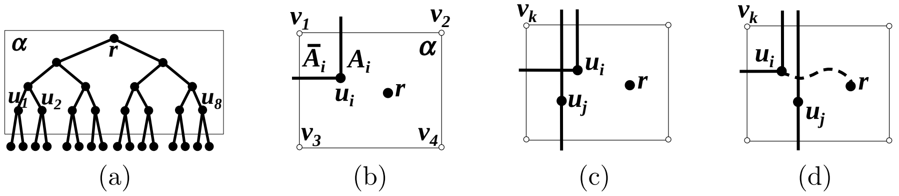


Figure 9.14: (a) A c-planar drawing of C. (b) Regions A_i and $\overline{A}_i$. (c) The edges connecting u_i to its children and the edges connecting u_j to its children cross. (d) $\overline{A}_i$ is enclosed inside $\overline{A}_j$.

The two edges connecting a node u_i and its children divide A in two regions A_i and $\overline{A}_i$ the first containing r and the second not. Since Γ is a straight-line orthogonal drawing, then both A_i and $\overline{A}_i$ contain at least one corner v_k (see Fig. 9.14 (b)). If two regions $\overline{A}_i$ and $\overline{A}_j$, with $i \neq j$, contain the same corner v_k, then either the edges connecting u_i to its children and the edges connecting u_j to its children cross (see Fig. 9.14 (c)), or $\overline{A}_i$ ($\overline{A}_j$) is enclosed inside $\overline{A}_j$ (resp. $\overline{A}_i$), and so the path connecting r and u_i (resp. connecting r and u_j) crosses one of the edges connecting u_j and its children (resp. connecting u_i and its children) (see Fig. 9.14 (d)).

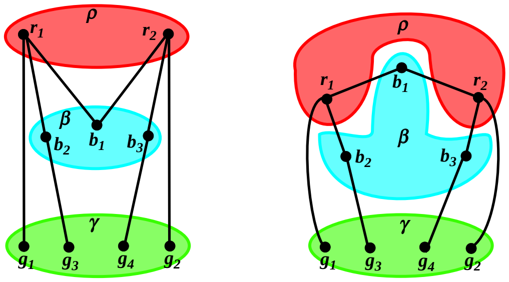
Since (i) each region $\overline{A}_i$ contains at least one corner v_k of A, (ii) any two regions $\overline{A}_i$ and $\overline{A}_j$, with $i \neq j$, cannot contain the same corner v_k, and (iii) there are four corners v_k and eight regions $\overline{A}_i$, then Γ cannot be an orthogonal straight-line R-drawing of C. □

9.4 R-Drawings and C-Drawings of Non-C-Connected C-Trees

In this section we consider R-drawings and C-drawings of non-c-connected clustered trees. We have that most of the positive results presented for c-connected trees are not achievable for non-c-connected trees, that seem to have the same area requirement of general clustered graphs. We begin by showing that upward drawings of non-c-connected clustered trees are generally not feasible.

Theorem 9.6 *There exists a non-c-connected c-planar clustered tree that does not admit any upward C-drawing.*

Proof: Consider the clustered tree $C = (G, T)$ defined as follows: G has root b_1, that has two children r_1 and r_2. Node r_1 (node r_2) has two children b_2 and g_1 (b_3 and g_2). Node b_2 (node b_3) has a child g_3 (resp. g_4). Vertices b_i, with $i \in \{1, 2, 3\}$, belong to cluster β, vertices r_i, with $i \in \{1, 2\}$, belong to cluster ρ, and vertices g_i, with $i \in \{1, 2, 3, 4\}$, belong to cluster γ. The inclusion tree T has root α that has three children β, ρ, and γ. It's easy to see that C is c-planar. Fig. 9.15 shows two c-planar drawings of C.



(a) (b)

Figure 9.15: (a) A non-upward c-planar drawing of C in which each cluster is represented by a convex region. (b) An upward c-planar drawing of C in which each cluster is represented by a non-convex region.

Suppose that an upward C-drawing Γ of C exists. Let b_x be the one between b_2 and b_3 that has minimum y-coordinate. Consider the horizontal strip S delimited by the horizontal lines through b_1 and through b_x. Let l_b be the segment connecting b_1 and b_x and let l_r be the segment connecting r_1 and r_2. Segment l_b divides S in two parts S_1 and S_2. The upwardness of Γ implies that $y(b_x) \leq y(r_1), y(r_2) \leq y(b_1)$. Hence, if r_1 and r_2 are not both in S_1 or both in S_2 segment l_b crosses segment l_r. However, the convexity of β and ρ implies that l_b and l_r belong entirely to the regions representing β and ρ in Γ, respectively. It follows that vertices r_1 and r_2 are both on the same of the parts S_1 and S_2 of S cut by l_b, otherwise β would cross ρ (see Fig. 9.16 (a)).

We claim that either there exists a node r_i, with $i \in \{1, 2\}$, that is enclosed inside a region R delimited by cluster β and by edges (b_j, r_k) (see Fig. 9.16 (b)), with $j \in \{1, 2, 3\}$, $k \in \{1, 2\}$, and $k \neq i$, or there exists a node b_i, with $i \in \{2, 3\}$ that is enclosed inside a region R delimited by cluster ρ and by edges (b_j, r_k) (see Fig. 9.16 (c)), with $j \in \{1, 2, 3\}$, $k \in \{1, 2\}$, and $j \neq i$.

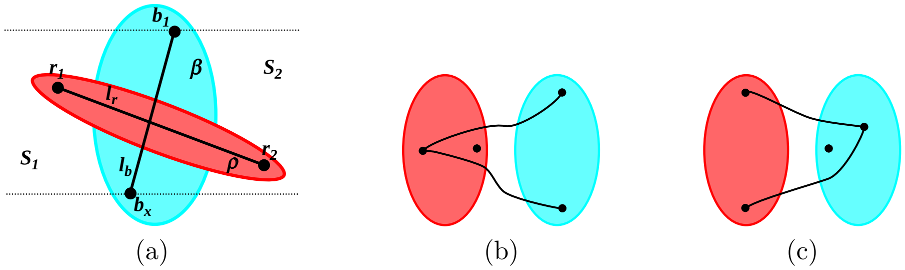


Figure 9.16: (a) Placing nodes r_1 and r_2 one in S_1 and one in S_2 leads to a region-region crossing. (b) A node r_i ($i \in \{1, 2\}$) enclosed inside the region delimited by β and by edges (b_j, r_k) ($j \in \{1, 2, 3\}$ and $k = 3 - i$). (c) A node b_i ($i \in \{2, 3\}$) enclosed inside the region delimited by cluster ρ and by edges (b_j, r_k) ($j \in \{1, 2, 3\}$, $j \neq i$ and $k \in \{1, 2\}$)

We consider two cases, depending on the y-coordinates of r_1 and r_2. If $y(r_1) \neq y(r_2)$, let r^* (resp. $\overline{r}$) be the one between r_1 and r_2 that has greater (resp. smaller) y-coordinate. Let h^* be the horizontal line through r^* and let p be any intersection point between $R(\beta)$ and h^*. Otherwise ($y(r_1) = y(r_2)$), let p be any intersection point between $R(\beta)$ and the line through r_1 and r_2. Let r^* (resp. $\overline{r}$) be the one between r_1 and r_2 that is closer (resp. farther) to p.

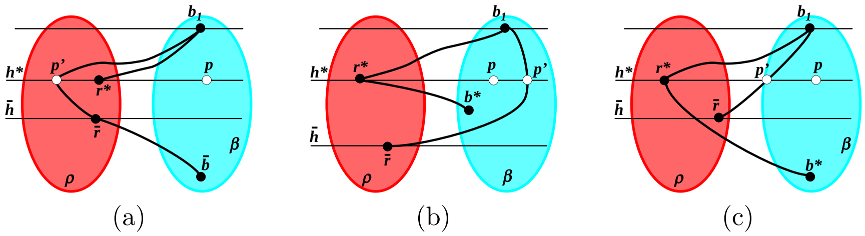


Figure 9.17: (a) Point p' lies outside β and r^* is closer than p' to p. (b) Point p' lies inside β or r^* is farther than p' to p, and edge (r^*, b^*) has no intersection with $\overline{h}$. (c) Point p' lies inside β or r^* is farther than p' to p, and edge (r^*, b^*) has intersection with $\overline{h}$.

Consider any intersection point p' between h^* and edge $(b_1, \overline{r})$.

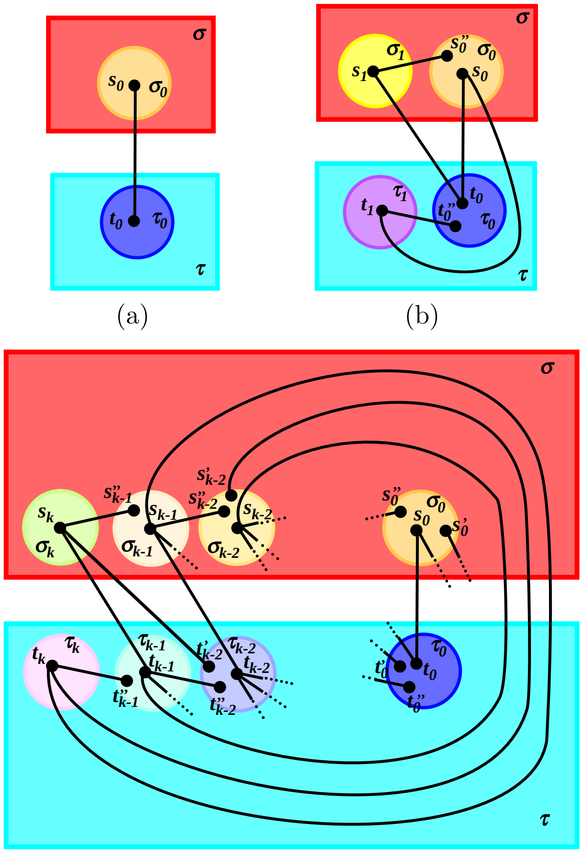
- If p' is outside β and r^* is closer than p' to p, then r^* is closed inside the region R delimited by cluster β, by edge $(b_1, \overline{r})$, and by edge $(\overline{r}, \overline{b})$ (Fig. 9.17 (a)).

- If p' is inside β or if r^* is farther than p' to p, then let $\overline{h}$ be the horizontal line through $\overline{r}$; we distinguish two cases:
  - If edge (r^*, b^*) has no intersection with $\overline{h}$, then b^* is closed inside the region R delimited by cluster ρ, by edge (b_1, r^*), and by edge $(b_1, \overline{r})$ (Fig. 9.17 (b)).
  - If edge (r^*, b^*) has intersection with $\overline{h}$, then $\overline{r}$ is closed inside the region R delimited by cluster β, by edge (b_1, r^*), and by edge (r^*, b^*) (Fig. 9.17 (c)).

Observe that every node b_i or r_j, with $i \in \{2, 3\}$ and $j \in \{1, 2\}$, has a child g_k, with $k \in \{1, 2, 3, 4\}$, belonging to cluster γ. Hence, the child g_k of the node b_i or r_j that is closed inside region R must lie inside R, as well, since placing g_k outside R would imply an edge crossing or an edge-region crossing. Moreover, the child g'_k of the node b_i that has minimum y-coordinate among the vertices of cluster β lies outside R, with $k \in \{3, 4\}$ and $i \in \{2, 3\}$. It follows that γ crosses region R, implying an edge-region crossing or a region-region crossing. □

Now we show that straight-line drawings of non-c-connected clustered trees may require exponential area. Let $C_k = (G_k, T_k)$ be the family of non-c-connected c-planar clustered trees inductively defined as follows.

- Clustered tree C_0 (see Fig. 9.18 (a)): Tree G_0 has vertices s_0 and t_0 and edge (s_0, t_0). The inclusion tree T_0 has a root node with two children σ and τ. Node σ (node τ) has one child σ_0 (resp. τ_0), where $s_0 \in V(\sigma_0)$ (resp. $t_0 \in V(\tau_0)$).
- Clustered tree C_1 (see Fig. 9.18 (b)): Tree G_1 is obtained from G_0 by adding vertices s_1, t_1, s''_0, and t''_0 and edges (s_1, t_0), (s_1, s''_0), (t_1, s_0) and (t_1, t''_0). The inclusion tree T_1 is obtained from T_0 by adding σ_1 to the children of σ and τ_1 to the children of τ, where $s''_0 \in V(\sigma_0)$, $s_1 \in V(\sigma_1)$, $t''_0 \in V(\tau_0)$, and $t_1 \in V(\tau_1)$.
- Clustered tree C_k, with $k > 1$ (see Fig. 9.18 (c)): Tree G_k is obtained from G_{k-1} by adding vertices s_k, t_k, s''_{k-1}, t''_{k-1}, s'_{k-2}, and t'_{k-2}, and edges (s_k, t_{k-1}), (s_k, s''_{k-1}), (t_k, s_{k-1}), (t_k, t''_{k-1}), (s_k, t'_{k-2}), and (t_k, s'_{k-2}). The inclusion tree T_k is obtained from T_{k-1} by adding σ_k to the children of



(a)

(b)

(c)

Figure 9.18: Inductive construction of the clustered tree C_k. (a) C_0. (b) C_1. (c) C_k.

σ and τ_k to the children of τ, where $s'_{k-2} \in V(\sigma_{k-2})$, $s''_{k-1} \in V(\sigma_{k-1})$, $s_k \in V(\sigma_k)$, $t'_{k-2} \in V(\tau_{k-2})$, $t''_{k-1} \in V(\tau_{k-1})$, and $t_k \in V(\tau_k)$.

It is easy to see (Fig. 9.19 (a)) that C_k is c-planar. Also, $G(\sigma)$, $G(\tau)$, $G(\sigma_i)$, and $G(\tau_i)$, with $i = 0, \ldots, k-1$, are not connected. For simplifying the notation, in the following we assume k is odd.

We have the following lemma (see Fig. 9.19 (b)):

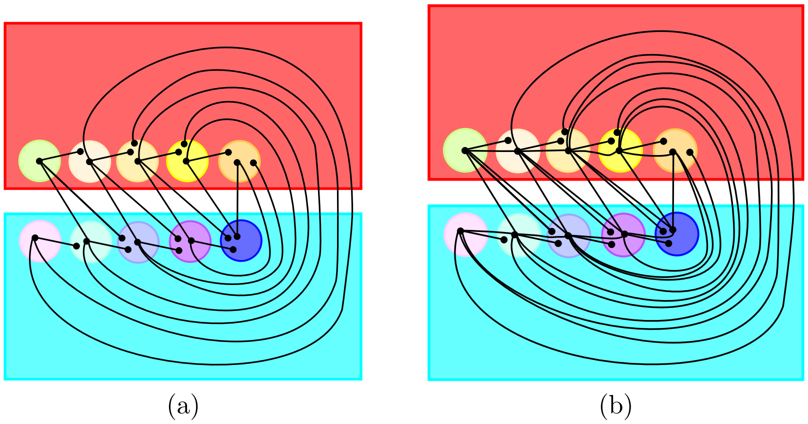


Figure 9.19: (a) A c-planar drawing of C_5. (b) A c-planar drawing of C_5 augmented as in Lemma 9.2.

Lemma 9.2 *In any c-planar drawing of C_k polygonal lines $l(s_0, s_1)$ connecting s_0 to s_1, $l(t_0, t_1)$ connecting t_0 to t_1 and, for $i = 2, \ldots, k$, $l(s_{i-1}, s_i)$ connecting s_{i-1} to s_i, $l(t_{i-1}, t_i)$ connecting t_{i-1} to t_i, $l(t_{i-2}, s_i)$ connecting t_{i-2} to s_i, and $l(s_{i-2}, t_i)$ connecting s_{i-2} to t_i can be drawn such that they do not cross between themselves, do not cross any edge of G_k, and: (1) $l(s_0, s_1)$ crosses only the border of clusters σ_0 and σ_1; (2) $l(t_0, t_1)$ crosses only the border of clusters τ_0 and τ_1; (3) $l(s_{i-1}, s_i)$ crosses only the border of clusters σ_{i-1} and σ_i; (4) $l(t_{i-1}, t_i)$ crosses only the border of clusters τ_{i-1} and τ_i; (5) $l(t_{i-2}, s_i)$ crosses only the border of clusters τ_{i-2}, τ, σ_i, and σ; and (6) $l(s_{i-2}, t_i)$ crosses only the border of clusters σ_{i-2}, σ, τ_i, and τ.*

Proof: We only show how to draw line $l(s_{i-1}, s_i)$; the other lines are drawn analogously. Consider any c-planar drawing Γ_k of C_k. Polygonal line $l(s_{i-1}, s_i)$ is composed of two parts: the first part is a segment between s_i and a point p_i arbitrarily close to s''_{i-1}; such a segment can be drawn arbitrarily close to segment (s_i, s''_{i-1}), so that it does not intersect any edge of G_k. Moreover, since (s_i, s''_{i-1}) crosses only the borders of clusters σ_{i-1} and σ_i, then (s_i, p_i) crosses only the borders of clusters σ_{i-1} and σ_i, as well. The second part of $l(s_{i-1}, s_i)$ is a polygonal line between p_i and s_{i-1}. Such points lie both inside the region representing σ_{i-1} in Γ_k. Since such a region contains only vertices s_{i-1}, s'_{i-1}, and s''_{i-1} that are not adjacent and contains entirely at most one polygonal

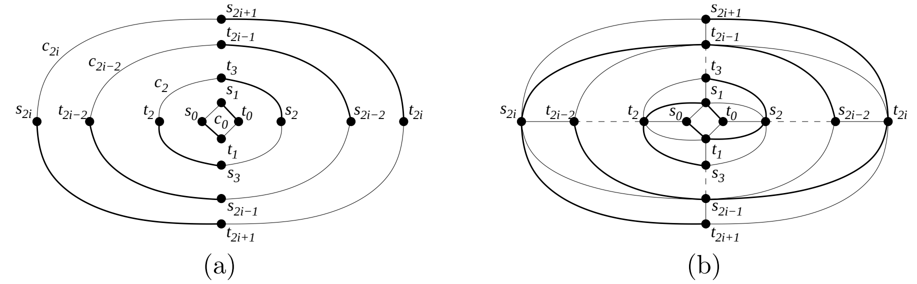


Figure 9.20: Graph G'_k. (a) Cycles c_{2i} and (b) their interconnections. Thick edges and thin edges distinguish among those edges that are common to G_k and G'_k and those edges that are not common to G_k and G'_k, respectively.

line between a point arbitrarily close to s'_{i-1} and s_{i-1} (such a line is part of $l(s_{i-1}, t_{i+1})$), then we can connect p_i and s_{i-1} with a polygonal line without creating crossings. □

Consider any straight-line C-drawing of clustered tree C_k. Augment it by the polygonal lines of Lemma 9.2. Now remove the vertices of G_k with apex "prime" or "double prime" and their incident edges. The resulting clustered graph $C'_k = (G'_k, T'_k)$ is exactly the one defined in [FCE95a] to prove an exponential area lower bound for straight-line C-drawings of clustered graphs.

More formally, graph G'_k is defined as follows (see Fig. 9.20). For $i = 0, 1, \ldots, \frac{k-1}{2}$, let c_{2i} be the simple cycle composed of edges (s_{2i}, s_{2i+1}), (t_{2i}, t_{2i+1}), (s_{2i}, t_{2i+1}), and (s_{2i+1}, t_{2i}). For $i = 0, 1, \ldots, \frac{k-3}{2}$, cycle c_{2i} is connected to c_{2i+2} by edges (s_{2i}, t_{2i+2}), (t_{2i}, s_{2i+2}), (s_{2i+1}, t_{2i+2}), (t_{2i+1}, s_{2i+2}), (s_{2i+1}, s_{2i+2}), (t_{2i+1}, t_{2i+2}), (s_{2i+1}, t_{2i+3}), and (t_{2i+1}, s_{2i+3}). The graph resulting from the connection of all the c_{2i} is G'_k. The inclusion tree T'_k is the subtree of T_k restricted to the vertices of G'_k.

In order to study the c-planar drawings of C_k, we study the ones of C'_k. Notice that Lemma 9.2 does not directly extend the exponential area lower bound from the straight-line C-drawings of C'_k to the ones of C_k, since, even if by Lemma 9.2 we can obtain a C-drawing of C'_k by augmenting any straight-line C-drawing of C_k, the edges necessary for such an augmentation (i.e., the polygonal lines of Lemma 9.2) are not forced to be drawn as straight lines.

Observe that, by the c-planarity of C_k and by Lemma 9.2, clustered tree C'_k is c-planar. Also, it is easy to see that G'_k is triconnected.

Since G'_k is triconnected, all the plane embeddings of G'_k differ only for the external face. Consider any face f of G'_k as external. Three cases are possible: (i) f coincides with c_0; (ii) f coincides with c_{k-1}; (iii) otherwise, let c_{2h} and c_{2h+2} be the cycles that contain the vertices of f. Selecting f as external face induces a nesting of the cycles c_{2i} of G'_k. In case (i) c_{2i+2} is contained into c_{2i}, for $i = 0, 1, \ldots, \frac{k-1}{2}$. In case (ii) c_{2i} is contained into c_{2i+2}, for $i = 0, 1, \ldots, \frac{k-1}{2}$. In case (iii) c_{2i} is contained into c_{2i+2}, for $i = 0, 1, \ldots, h$, and c_{2i+2} is contained into c_{2i}, for $i = h+1, h+2, \ldots, \frac{k-1}{2}$. In all three cases there is a nesting composed of at least $\lceil (k-1)/4 \rceil$ cycles. The area lower bound for C'_k will be obtained by considering the area requirement of such a nesting.

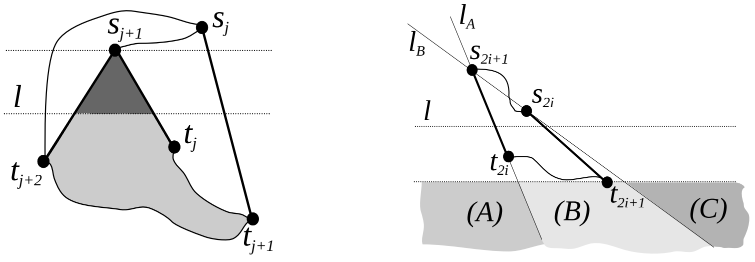
The following lemma is a generalization of Theorem 4 in [FCE95a]. In that paper the drawings of C'_k are studied where all the edges are straight-lines, while in the following lemma only the edges of G'_k that are also edges of G_k are required to be straight.

Lemma 9.3 *Any c-planar drawing of C'_k such that the edges of G_k are straight-line segments and the clusters are represented by convex polygons requires $\Omega(b^n)$ area, with $b > 1$.*

Proof: Consider any c-planar drawing Γ' of C'_k, in which the edges of G_k are straight-line segments and the clusters are represented by convex polygons. As already discussed, in Γ' there is a nesting of at least $\lceil (k-1)/4 \rceil$ cycles. Rename the vertices of G'_k (and of G_k) according to such a nesting. Namely, call c_0 the most nested cycle, and s_0, t_0, s_1, and t_1 its vertices. Call c_2 the cycle surrounding c_0, and s_2, t_2, s_3, and t_3 its vertices, etc. The outermost cycle is denoted by c_{2d}. Observe that $d = \Omega(k) = \Omega(n)$.

Let Γ'_{2i} denote the part of Γ' embedded inside c_{2i}, including such a cycle. Notice that, among the edges (s_{2i}, s_{2i+1}), (t_{2i}, t_{2i+1}), (s_{2i}, t_{2i+1}), and (t_{2i}, s_{2i+1}) composing cycle c_{2i}, only edges (s_{2i}, t_{2i+1}) and (t_{2i}, s_{2i+1}) are necessarily straight-lines in Γ'. Because of the convexity of the regions $R(\sigma)$ and $R(\tau)$ representing σ and τ, respectively, there exists a line l in Γ' separating $R(\sigma)$ and $R(\tau)$. Suppose w.l.o.g. that l is horizontal and that $R(\sigma)$ is above $R(\tau)$. Denote by H^+ and by H^- the half-planes above and below l, respectively. We argue that the area of Γ'_{2i+2} is at least twice the one of Γ'_{2i}, for $0 \leq i \leq d-1$. The thesis follows from this argument.

First, we show that $y(s_j) < y(s_{j+1})$, $0 \leq j \leq 2d-2$. Suppose, for a contradiction, that $y(s_j) \geq y(s_{j+1})$ (Fig. 9.21 (a)). We claim that s_j is outside the region R_j delimited by edges (s_{j+1}, t_j), (s_{j+1}, t_{j+2}), (t_{j+1}, t_j) and (t_{j+1}, t_{j+2}). Namely, s_j lies in H^+, hence if R_j contains s_j in its interior, then the inter-



(a) (b)

Figure 9.21: Illustrations for the proof of Lemma 9.3. (a) $y(s_{2i}) \geq y(s_{2i+1})$. The grey region (both the dark and the light grey) is R_j. The dark grey region is I_j. (b) Possible placements for vertex t_{2i+2}. Region (A), Region (B), and Region (C) have different shades of grey.

section I_j between R_j and H^+ (dark grey in Fig. 9.21 (a)) contains s_j in its interior, as well. However, since edges (s_{j+1}, t_j) and (s_{j+1}, t_{j+2}) are straight-lines and since edges (t_{j+1}, t_j) and (t_{j+1}, t_{j+2}) cannot intersect l because of the supposed c-planarity of Γ', I_j is a triangle whose uppermost vertex is s_{j+1}. Since $y(s_j) \geq y(s_{j+1})$, I_j cannot contain s_j in its interior and R_j cannot contain s_j in its interior, as well. Two cases are possible: j is even or j is odd (recall that cycles c_i's have an even index). Suppose j is even. Since s_j is outside R_j, cycle c_{j+2} cannot be external to cycle c_j, contradicting the assumption that cycle c_{j+2} is drawn externally with respect to cycle c_j. Suppose j is odd. Since s_j is outside R_j, cycle c_{j+1} cannot be external to cycle c_{j-1}, contradicting the assumption that cycle c_{j+1} is drawn externally with respect to cycle c_{j-1}.

An analogous proof shows that $y(t_j) > y(t_{j+1})$, for $0 \leq j \leq 2d-2$.

Consider the placement of vertex t_{2i+2} in Γ'. Let l_A be the line through t_{2i} and s_{2i+1} and let l_B be the line through s_{2i} and s_{2i+1}. By the above discussion, $y(t_{2i+2}) < y(t_{2i+1})$ holds. Hence, vertex t_{2i+2} can only lie in one of the regions (A), (B), and (C) defined below and outside the polygon delimited by cycle c_{2i} (composed by edges (s_{2i}, t_{2i+1}), (s_{2i}, s_{2i+1}), (t_{2i}, t_{2i+1}), and (t_{2i}, s_{2i+1})). (See Fig. 9.21 (b)). Region (A), that is the intersection region between the half-plane delimited by l_A and not including s_{2i}, and the half-plane $y < y(t_{2i+1})$. Region (B), that is the intersection region between the half-plane delimited by l_A and including s_{2i}, the half-plane delimited by l_B and including t_{2i}, and the half-plane $y < y(t_{2i+1})$. Region (C), that is the intersection region between the half-plane delimited by l_B and not including t_{2i}, and the half-plane $y < y(t_{2i+1})$.

If vertex t_{2i+2} is placed inside Region (A), then vertex t_{2i} is enclosed inside the cycle $\mathcal{C}$ composed of edges (s_{2i+1}, s_{2i}), (s_{2i}, t_{2i+1}), (t_{2i+1}, t_{2i+2}), and (t_{2i+2}, s_{2i+1}) (see Fig. 9.22 (a)). Since vertex s_{2i+2} has to be connected to vertex t_{2i}, then, by the supposed planarity of Γ', s_{2i+2} is enclosed inside $\mathcal{C}$, as well. However, this contradicts the assumption that cycle c_{2i+2} is external with respect to cycle c_{2i}.

If vertex t_{2i+2} is placed inside Region (B) then Γ' has a crossing. Namely, edge (s_{2i+1}, t_{2i+2}) is a straight-line segment in Γ'. Hence, if vertex t_{2i+2} is placed inside Region (B) edge (s_{2i+1}, t_{2i+2}) crosses either edge (t_{2i+1}, t_{2i}) or edge (s_{2i}, t_{2i+1}). In fact such edges separate vertex s_{2i+1} from Region (B) (see Fig. 9.22 (b)).

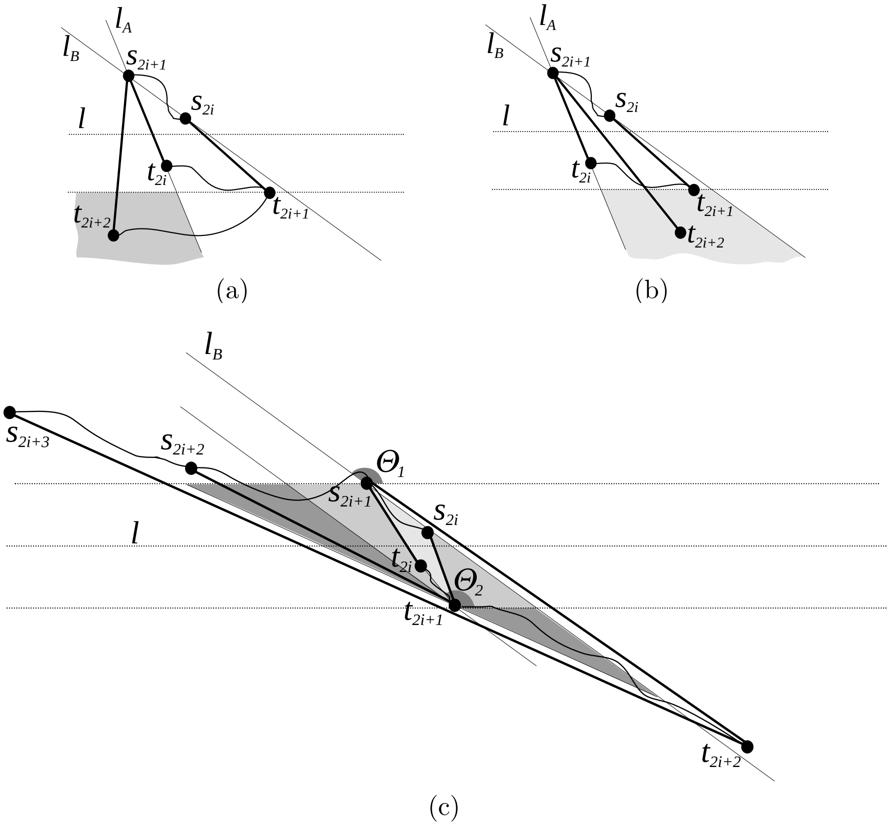
Hence, we have that Region (C) is the only possible placement of t_{2i+2}. This geometric constraint on the placement of t_{2i+2} is exactly the same that was exploited in [DTT92] to prove an exponential area lower bound for straight-line upward drawings of planar directed graphs.

Let Θ_1 be the angle formed by line l_B and by the x-axis and let Θ_2 be the angle formed by the line through t_{2i} and t_{2i+1} and by the x-axis. In [DTT92] it is shown that (suppose that $\Theta_1 \geq \Theta_2$ and see Fig. 9.22 (c)): (i) the parallelogram P^* delimited by the horizontal lines through s_{2i+1} and t_{2i+1}, by l_B and by the line through t_{2i+1} parallel to l_B has area at least twice the area of the cycle composed of edges (t_{2i}, s_{2i+1}), (s_{2i+1}, t_{2i+1}), (t_{2i+1}, s_{2i}), and (s_{2i}, t_{2i}); (ii) the triangle T^* delimited by the horizontal line through s_{2i+1}, by l_B and by the line through t_{2i+1} parallel to edge (t_{2i+2}, s_{2i+3}) contains P; (iii) the drawing of the cycle composed of edges (t_{2i+2}, s_{2i+3}), (s_{2i+3}, s_{2i+2}), (s_{2i+2}, t_{2i+3}), and (t_{2i+3}, t_{2i+2}) contains T^*. Properties symmetric to properties (i), (ii), and (iii) hold if $\Theta_1 < \Theta_2$.

Such arguments straightforwardly apply here (by suitably replacing the area of the drawing with the convex-hull area of the vertices) and this concludes the proof. □

Lemma 9.4 *If there exists a straight-line C-drawing of C_k with area A, then there exists a c-planar drawing of C'_k such that the edges of G_k are straight-line segments, the clusters are represented by convex polygons, and the area is less or equal than A.*

Proof: Consider any C-drawing of C_k with area A. It can be augmented without increasing the area by inserting the polygonal lines of Lemma 9.2, still remaining c-planar. At this point the vertices that do not belong to G'_k and



(a)

(b)

(c)

Figure 9.22: (a) If vertex t_{2i+2} is placed inside Region (A), then the embedding of G'_k changes. (b) If vertex t_{2i+2} is placed inside Region (B), then Γ' has a crossing. (c) Placement of vertex t_{2i+2} inside Region (C). Parallelogram P^* is composed of the regions colored by light shades of grey. Triangle T^* is composed of all regions colored by grey.

their incident edges can be removed obtaining a c-planar drawing of C'_k with area less or equal than A. □

From the above lemmas we have:

Theorem 9.7 *There exists an n-vertex non-c-connected c-planar clustered tree* *requiring $\Omega(b^n)$ area in any straight-line C-drawing, with $b > 1$.*

The above lower bound is matched by an exponential upper bound. Namely, one can augment the non-c-connected c-planar clustered tree in a c-connected c-planar clustered graph, that admits an exponential area C-drawing, by the results in [EFLN06]. If we relax the straight-line constraints, then better results can be obtained:

Theorem 9.8 *There exists an algorithm that computes an order-preserving* 2*-bends poly-line R-drawing requiring $\Theta(n^2)$ area of every non-c-connected c-planar clustered tree.*

Proof: The proof is strongly based on the results of Eades, Feng, and Nagamochi in [EFN99]. Namely, in [EFN99] an algorithm is shown for computing an orthogonal c-planar drawing of a clustered graph $C = (G, T)$ such that G has maximum degree 4. As a first step, such an algorithm computes an $O(n^2)$ area c-planar drawing of C where the drawing of G is a visibility representation and where each cluster is drawn as a rectangle having sides parallel to the axes and having corners with integer coordinates. As noticed in [EFN99], a visibility representation with the above described features can be constructed for a clustered graph whichever is the maximum degree of its underlying graph.

We use such results as follows: (i) according to Theorem 2 of [FCE95b], augment the input clustered tree $C = (G, T)$ to a c-connected c-planar clustered graph $C' = (G', T)$ with the same number of vertices and with some extra dummy edges; (ii) compute a visibility representation Γ' of C' by using the algorithm in [EFN99]; (iii) turn Γ' into a poly-line drawing of C', using the techniques presented by Di Battista and Tamassia in [DT88] and sketched in Sect. 2.3; (iv) remove the dummy edges from Γ', obtaining an R-drawing Γ of the clustered tree C.

It is easy to observe that turning a visibility drawing into a poly-line drawing preserves the c-planarity of the drawing. Moreover, the area of Γ is quadratic and, by Lemma 9.1, such a bound is optimal. Notice that at most two bends per edge are introduced by the algorithm.

Concerning the running time of the described algorithm, it has been observed in [EFN99] that, supposing the c-connected clustered graph C' to be given, then a c-planar visibility representation Γ' of C' can be computed in linear time. Further, it's easy to see that turning the visibility representation into a poly-line drawing can be performed in linear time, as well. Hence the total running time is linear if C' is given. If not, then the running time can not be assumed neither polynomial, since the complexity of providing a c-connected c-planar clustered graph containing a non-c-connected clustered tree as a subgraph is unknown, as far as we know (see also Chapter 10). □

9.5 NC-Drawings of C-Connected and Non-C-Connected C-Trees

In this section we consider c-planar drawings of clustered trees, assuming that each cluster is drawn as a simple, potentially non-convex, lattice polygon. We show that polynomial area is sufficient for strictly upward order-preserving straight-line NC-drawings of c-connected clustered trees. Notice that in the same drawing convention whether R- and C-drawings require polynomial or exponential area is open.

We show an inductive algorithm to construct a strictly upward order-preserving straight-line NC-drawing of a c-connected clustered tree $C = (G, T)$. Let r be the root of G and let $G(r_1)$, $G(r_2)$, …, $G(r_k)$ be the subtrees of G rooted at the children r_1, r_2, …, r_k of r, respectively. Suppose that, for each $C_i = (G(r_i), T_i)$, where $1 \leq i \leq k$ and where T_i is the subtree of T induced by the clusters containing at least one vertex of $G(r_i)$, a strictly upward NC-drawing Γ_i can be constructed. Suppose also that each cluster μ of T_i is represented in Γ_i by a polygonal line composed by four parts (see Fig. 9.23): A horizontal segment $T(\mu)$ delimiting the top side of the cluster and lying on the line $y = y_T(\mu)$, two vertical segments $L(\mu)$ and $R(\mu)$ delimiting the left and right sides of the cluster and lying on the lines $x = x_L(\mu)$ and $x = x_R(\mu)$, respectively, and one polygonal line $B(\mu)$ monotonically increasing in the x-direction delimiting the bottom side of the cluster.

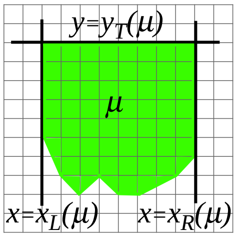


Figure 9.23: Shape of a cluster in the algorithm to construct strictly upward order-preserving straight-line NC-drawings of c-connected clustered trees.

The above induction hypothesis is easily verified in the base case. Namely, if $G(r)$ has only one vertex v, draw it on a grid point. The clusters containing v are drawn as squares enclosing each other.

Now, suppose that $G(r)$ has more than one node. Inductively assume to have an NC-drawing Γ_i of each C_i. For each i such that $1 \leq i \leq k$, consider the set V_i of vertices of $G(r_i)$ and the set S_i of clusters belonging to T_i that do not contain r. Let $x_L(\Gamma_i) = \min_{v \in V_i, \mu \in S_i}\{x(v), x_L(\mu)\}$, $x_R(\Gamma_i) = \max_{v \in V_i, \mu \in S_i}\{x(v), x_R(\mu)\}$, and $y_T(\Gamma_i) = \max_{\mu \in S_i}\{y(r_i), y_T(\mu)\}$. For each i such that $1 \leq i \leq k$, remove the part of Γ_i (observe that it does not contain vertices of G) that is inside one of the three half-planes $x < x_L(\Gamma_i)$, $x > x_R(\Gamma_i)$, and $y > y_T(\Gamma_i)$. This gives us partial drawings Γ'_i of all the C_i's, where the notations $x_L(\Gamma)$, $x_R(\Gamma)$, and $y_T(\Gamma)$ are extended to $x_L(\Gamma')$, $x_R(\Gamma')$, and $y_T(\Gamma')$, respectively, in the obvious way.

Place the Γ'_i's one beside the other, with $x_L(\Gamma'_{i+1}) = x_R(\Gamma'_i) + 1$, and so that all the r_i's lie on the same horizontal line h. Place r $2n^2$ units above and on the same vertical line of r_1. Draw straight-line edges between r and its children. Consider the clusters $\mu_1, \mu_2, \dots, \mu_l$ containing r ordered so that μ_j is a sub-cluster of μ_{j+1}, for $1 \leq j < l$. In the following we show how to draw each cluster μ_j:

- Draw $T(\mu_j)$ as a horizontal segment between points $(x_L(\Gamma'_1) - j, y(r) + j)$ and $(x_R(\Gamma'_k) + j, y(r) + j)$.

- Draw $L(\mu_j)$ as a vertical segment between points $(x_L(\Gamma'_1) - j, y(r) + j)$ and $(x_L(\Gamma'_1) - j, y_T(\Gamma'_1) + l - j + 1)$.

- Draw $R(\mu_j)$ as a vertical segment between endpoints $(x_R(\Gamma'_k) + j, y(r) + j)$ and $(x_R(\Gamma'_k) + j, y_T(\Gamma'_k) + l - j + 1)$.

- We show how to draw $B(\mu_j)$. For each Γ'_i and each μ_j such that T_i does not contain μ_j, with $1 \leq i \leq k$ and $1 \leq j \leq l$, draw a horizontal segment between points $(x_L(\Gamma'_i), y_T(\Gamma'_i) + l - j + 1)$ and $(x_R(\Gamma'_i), y_T(\Gamma'_i) + l - j + 1)$. Notice that now for each Γ'_i and each μ_j the part of $B(\mu_j)$ between x-coordinates $x_L(\Gamma'_i)$ and $x_R(\Gamma'_i)$ has been drawn. We call that part $B(\Gamma'_i, \mu_j)$. For each pair $(\Gamma'_i, \Gamma'_{i+1})$ and each μ_j, with $1 \leq i < k$ and $1 \leq j \leq l$, connect $B(\Gamma'_i, \mu_j)$ and $B(\Gamma'_{i+1}, \mu_j)$ by a segment between the rightmost point of $B(\Gamma'_i, \mu_j)$ and the leftmost point of $B(\Gamma'_{i+1}, \mu_j)$. Polygonal line $B(\mu_j)$ is completed by a segment connecting $(x_L(\Gamma'_1) - j, y_T(\Gamma'_1) + l - j + 1)$ and the leftmost point of $B(\Gamma'_1, \mu_j)$ and a segment connecting $(x_R(\Gamma'_k) + j, y_T(\Gamma'_k) + l - j + 1)$ and the rightmost point of $B(\Gamma'_k, \mu_j)$.

An example of application of the algorithm is shown in Fig. 9.24. We obtain the following:

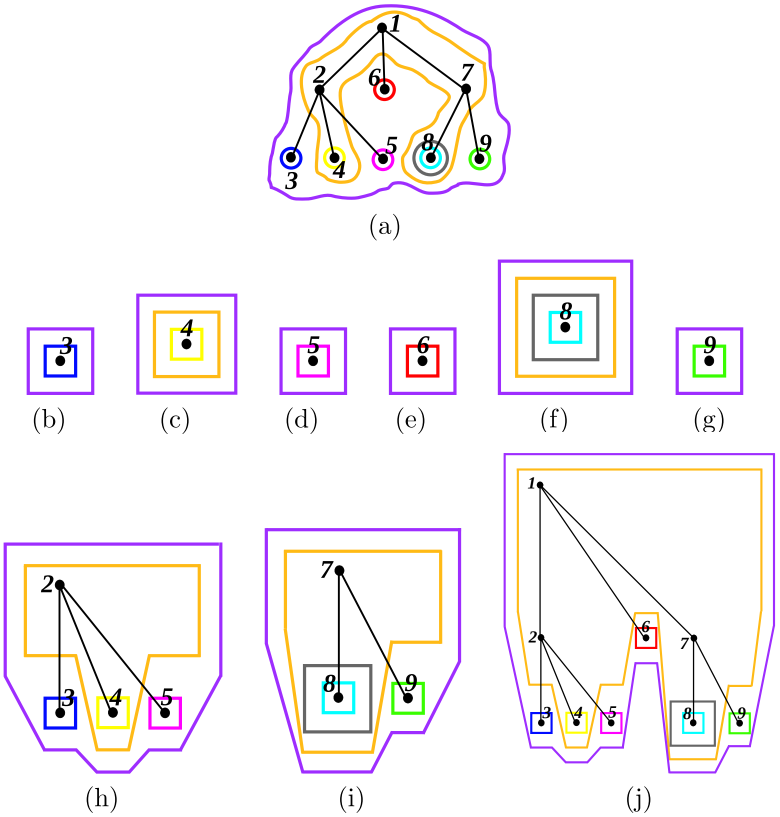


Figure 9.24: (a) An ordered c-connected clustered tree $C = (G, T)$. (b)–(j) Application of the algorithm for constructing a strictly-upward order-preserving straight-line NC-drawing of C.

Theorem 9.9 *For every c-connected clustered tree there exists a strictly upward order-preserving straight-line NC-drawing in $O(n^4)$ area.*

Proof: Let $C = (G, T)$ be a c-connected clustered tree. Apply the algorithm described in this section with C as an input. It's easy to see that the obtained drawing Γ is strictly upward, order-preserving, planar and straight-line. An easy inductive argument can be used to prove the c-planarity of Γ. In particular, the absence of edge-region crossings is guaranteed by the high value of the slopes of the edges of G.

Concerning the area bound, it's easy to see that the height of the drawing increases by $O(n^2)$ at each inductive step; since there are $O(n)$ steps the height of the drawing is $O(n^3)$. Concerning the width, the observation that for each vertical line there is either a vertex or one of the two lateral sides enclosing a cluster leads to a $O(n)$ width. □

We conclude the section with the following theorem.

Theorem 9.10 *For every c-connected binary clustered tree $C = (G, T)$ there exists a straight-line orthogonal upward NC-drawing with $O(n^3 \log n)$ area.*

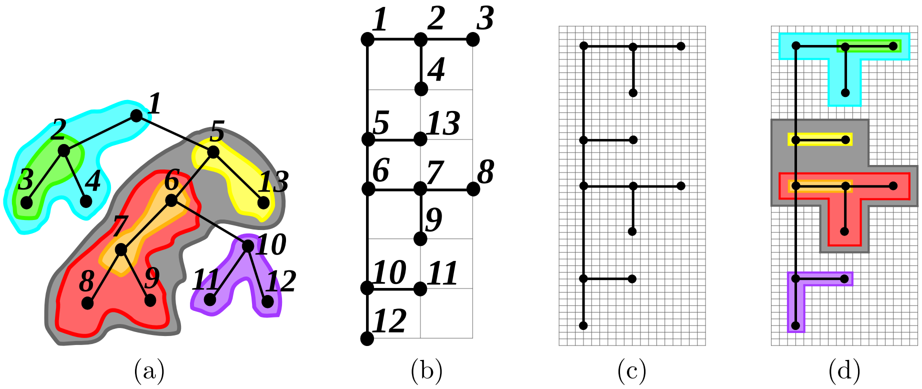


Figure 9.25: (a) A c-connected binary clustered tree $C = (G, T)$. Notice that $h(T) = 4$. (b) An hv-drawing Γ of G [CBP92] with $O(n)$ height and $O(\log n)$ width. (c) Augmenting the grid of Γ. Notice that $2(h(T) - 1) = 6$. (d) The NC-drawing of C constructed on Γ.

Proof: Let $C = (G, T)$ be a c-connected binary clustered tree (see Fig. 9.25 (a)). Construct an hv-drawing Γ of G with $O(n)$ height and $O(\log n)$ width, by the algorithm in [CBP92] (see Fig 9.25 (b)). Augment the grid by inserting, for each column of Γ (for each row of Γ), $2(h(T) - 1)$ vertical grid lines (resp.

$2(h(T)-1)$ horizontal grid lines), where $h(T)$ is the number of edges in the longest path from the root to a leaf in T (see Fig 9.25 (c)). Such lines are used to draw each cluster μ as an orthogonal non-convex polygon $P(\mu)$. This can be easily done by proceeding bottom-up on the inclusion tree; each cluster surrounds the already drawn clusters and the part of the tree that it contains (see Fig 9.25 (d)). □

9.6 Conclusions and Open Problems

In this chapter we dealt with the problem of obtaining minimum area c-planar drawings of clustered graphs whose underlying graph is required to be a tree.

Tables 9.1 and 9.2 summarize area bounds proved for the different drawing standards considered. Tables 9.1 and 9.2 are also a reference point for classifying open problems. They correspond to question marks, to cells where upper and lower bound do not match, and to cells where a drawing is in general not feasible. Such latter cells open the problem of recognizing the clustered trees that have a feasible drawing. We would like to explicitly mention two of such open problems that seem especially interesting.

We have shown that polynomial area is sufficient for obtaining c-planar drawings in most of the drawing's styles, contrasting with the result presented in [EFLN06, Fen97], where it is shown that c-planar straight-line drawings of c-connected clustered graphs generally require exponential area. However, the following problem remains open:

Open Problem 9.1 *Which are the asymptotic bounds for the area requirements of (strictly-upward) order-preserving straight-line drawings of c-connected clustered trees?*

We notice that an $O(n^2)$ area upper bound for such a problem would imply most of our positive results on c-connected trees.

Concerning non-c-connected clustered trees, we have shown that straight-line drawings could require as much area as that required by clustered graphs, hence nothing is earned by requiring the underlying graph to be a tree. Even if the mentioned lower bound is matched by an upper bound in the case of C-drawings, whether straight-line R-drawings of non-c-connected clustered trees always exist is still an open question. Such a problem is, as far as we know, open also for c-connected clustered graphs.

Open Problem 9.2 *Does every non-c-connected clustered tree (every c-connected clustered graph) admit a straight-line R-drawing?*

Chapter 10

C-Planarity of Embedded Flat Clustered Graphs with Small Faces

In this chapter[1] we consider the problem of efficiently testing the *c*-planarity of clustered graphs, probably the most studied topic in the Graph Drawing community during the last ten years. We consider embedded clustered graphs, that is, clustered graphs for which the planar embedding of the underlying graph is fixed, and we give a first contribution towards designing a polynomial-time testing algorithm for such graphs. Namely, we characterize *c*-planar embedded flat clustered graphs with at most five vertices per face and give an efficient testing algorithm for such graphs. The results are based on a more general methodology that sheds new light on the *c*-planarity testing problem.

10.1 Introduction

Determining the computational complexity of the *c*-planarity testing for clustered graphs is one of the main Graph Drawing challenges. The problem takes as an input a clustered graph $C = (G, T)$ and asks whether C is *c*-planar or not. Despite all the research efforts spent, only for restricted families of clustered graphs polynomial-time testing algorithms have been found, and the general problem is still open.

[1] The contents of this chapter are a joint work with Giuseppe Di Battista, appeared in [BF07] and submitted to journal.

A brief survey on the problem of testing the *c*-planarity of clustered graphs can be found in [CB05]. The classes of clustered graphs for which the problem is known to be polynomial-time solvable are the following:

- *c-connected clustered graphs*, in which each cluster induces a connected subgraph of the underlying graph; the first polynomial-time testing algorithm for this class has been presented in [FCE95b]; a linear-time testing algorithm has been proposed in [Dah98]; a characterization and a consequent linear-time testing algorithm have been shown in [CBF$^+$08].

- *completely connected clustered graphs*, that are *c*-connected clustered graphs such that the complement of the subgraph induced by each cluster is connected; an elegant characterization for this class is shown in [CW06].

- *almost connected clustered graphs*, in which either all the nodes corresponding to non-connected clusters are on the same path in the cluster hierarchy, or for each non-connected cluster its parent and all its siblings are connected [GJL$^+$02].

- *extrovert clustered graphs*, a generalization of *c*-connected clustered graphs with special restrictions on the cluster hierarchy [GLS05].

- *cycles of clusters*, in which the hierarchy is flat, the underlying graph is a simple cycle, and the clusters are arranged in a cycle [CDPP05a].

- *clustered cycles*, that are clustered graphs in which the hierarchy is flat, the underlying graph is a simple cycle, and the clusters are arranged into an embedded plane graph [CDPP05b].

- *k-rib Eulerian graphs*, that are clustered graphs in which the underlying graph is Eulerian, and the graph can be obtained from a 3-connected graph on k vertices, for some constant k, by multiplying and subdividing some edges [JKK$^+$07].

- *clustered graphs with four outgoing edges*, that are clustered graphs in which every cluster has at most four edges that have one end-vertex inside the cluster and the other end-vertex outside the cluster [SJTV08].

- *embedded clustered graphs with two components for each cluster*, that are clustered graphs in which every cluster induces at most two connected components [JJKL08].

Let C be a clustered graph. Suppose that C is *embedded*, that is, the combinatorial embedding of the underlying graph of C is fixed (then an embedded clustered graph is c-planar if it admits a c-planar drawing in which the embedding of G is preserved). Is testing the c-planarity of C easier than in the variable embedding setting? This question is motivated by the existence of many Graph Drawing problems on planar graphs that are in general NP-hard and that become polynomial-time solvable if the embedding is fixed. Testing if a graph admits an orthogonal planar drawing with at most k bends [Tam87, GT01] or if a graph admits an upward planar drawing [BDLM94, GT01] are examples of such problems.

In this chapter we give a first contribution towards answering the above question. Namely, we characterize c-planar embedded flat clustered graphs with at most five vertices per face and give an efficient testing algorithm for such graphs. A *flat clustered graph* (G, T) is such that in any path from the root to a leaf of T there are at most three nodes (the clustered graph in Fig. 10.1 is flat). Hence, in a flat clustered graph the only non-trivial clusters are all and only the children of the root. Hence, when referring to a flat clustered graph, we call *clusters* only the children of the root. Also, given a vertex v of the underlying graph we say that the *cluster of* v is its parent in T.

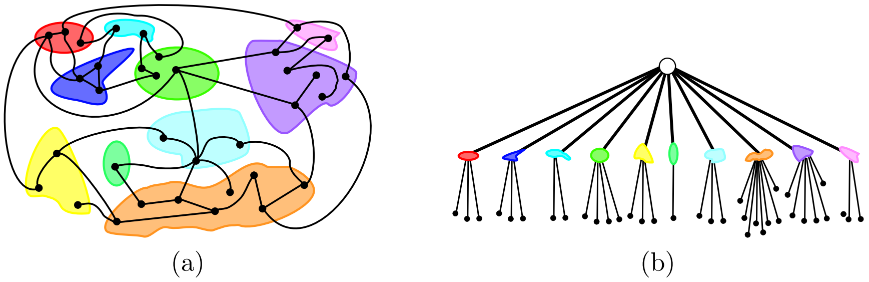


Figure 10.1: (a) A flat clustered graph $C(G, T)$. (b) The inclusion tree T of C has height three.

Our approach is to look for an augmentation that adds to the embedded underlying graph extra edges such that the resulting clustered graph is c-connected and c-planar. We call *candidate saturating edges* those edges that are potential candidates for the augmentation. Two of such edges have a *conflict* if using both of them in the augmentation causes a crossing. We present a characterization for *single-conflict embedded flat clustered graphs*, that are embedded flat clustered graphs such that each candidate saturating edge has a conflict with at most one other candidate saturating edge. The characterization and the algorithm for embedded flat clustered graphs with at most five vertices per face are a consequence of such a more general result.

We remark that in the following we deal both with biconnected and with simply connected embedded planar graphs. In the former case, the "number of vertices in a face" is trivially defined as the number of vertices incident to the face, while in the latter case is meant to be the number of occurrences of vertices on the border of the face.

A slightly weaker result, namely a quadratic time algorithm for c-planarity on 3-connected graphs with faces of size at most four, was independently discovered by Jelinkova et al. in [JKK$^+$07].

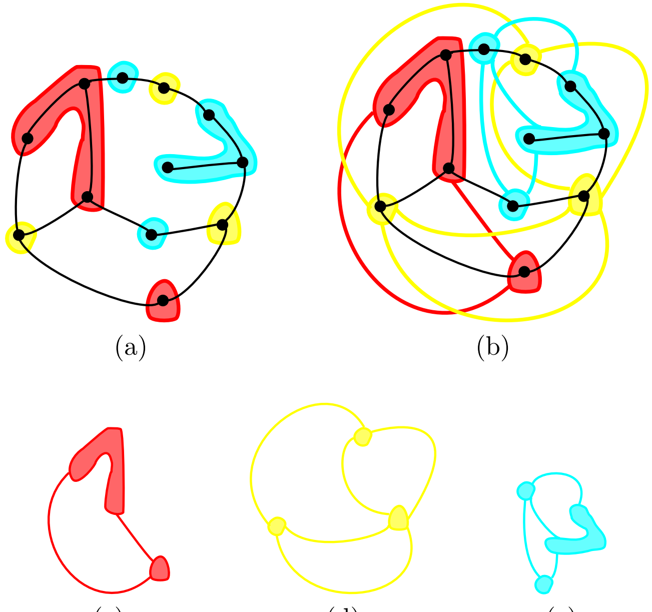
The rest of the chapter is organized as follows. In Sect. 10.2 we introduce notions about augmentations and saturations of clustered graphs; in Sect. 10.3 we characterize *c*-planar single-conflict embedded flat clustered graphs and *c*-planar embedded flat clustered graphs with at most five vertices per face; in Sect. 10.4 we present a linear time and space *c*-planarity testing algorithm; finally, in Sect. 10.5 we conclude and present some open problems.

We remark that the contents of this chapter (characterizations of the *c*-planar clustered graphs and algorithms for testing the *c*-planarity of clustered graphs) are not in the mainstream of this thesis (that is about the construction of geometric representations of graphs with small area). However, we still decided to present the results of this chapter in the thesis for two main reasons: (1) the *c*-planarity testing has a strong relation with the topics of Chapter 9, namely any algorithm for constructing *c*-planar drawings of clustered graphs should first verify the *c*-planarity of the input graph; (2) the problem of testing the *c*-planarity of clustered graphs is beautiful and it is probably the problem that is attracting the greatest research efforts in the Graph Drawing community.

10.2 Augmentations and Saturations

In this section we study augmentations and saturations of non-*c*-connected clustered graphs.

Consider an embedded flat clustered graph $C(G,T)$. For each face f of G a set of *candidate saturating edges* is defined as follows: Let O be the clockwise circular order of the vertices on the border of f. Subdivide such vertices into subsets such that each subset V_i contains a maximal sequence of consecutive vertices in O belonging to the same cluster. Introduce a candidate saturating edge for each two subsets $V_i \neq V_j$ such that (i) V_i and V_j contain vertices of the same cluster μ_k and (ii) V_i and V_j are in different connected components of $G(\mu_k)$. Candidate saturating edges represent edges that can be added to the embedded clustered graph to make the subgraph induced by each cluster connected (see Fig. 10.2 (a) and 10.2 (b)).


(a)

(b)

(c)

(d)

(e)

Figure 10.2: (a) An embedded flat clustered graph C. Different clusters have different colors. The connected components of each cluster are inside simple connected regions having the color of the cluster. (b) Clustered graph C and its candidate saturating edges. Candidate saturating edges of each cluster have the same color of the cluster. (c)–(d)–(e) Multigraphs $\mathcal{G}_i$ for C. The vertices of $\mathcal{G}_i$ are the connected components of $G(\mu_i)$ and the edges of $\mathcal{G}_i$ are the candidate saturating edges connecting such components.

For a cluster μ_i of T we define $\mathcal{G}_i$ as the embedded multigraph whose vertices are the connected components of $G(\mu_i)$ and whose edges are the candidate saturating edges connecting such components. The embedding of $\mathcal{G}_i$ is given by the order of the faces around the vertices of G (Fig. 10.2 (c), 10.2 (d), and 10.2 (e)). Observe that $\mathcal{G}_i$ does not have self-loops and is, in general, non-planar. However, possible crossings are only between edges introduced in the same face of G.

Two candidate saturating edges e_1, joining connected components $G_1(\mu_i)$ and $G_2(\mu_i)$ of $G(\mu_i)$, and e_2, joining connected components $G_1(\mu_j)$ and $G_2(\mu_j)$ of $G(\mu_j)$, with $\mu_i \neq \mu_j$ and with e_1 and e_2 in the same face f of G, have a *conflict* if $G_1(\mu_i)$, $G_1(\mu_j)$, $G_2(\mu_i)$, and $G_2(\mu_j)$ appear in this order around the border of f. Informally speaking, two candidate saturating edges have a conflict if adding both of them to the clustered graph causes a crossing.

The following theorem shows the role of the candidate saturating edges of a flat embedded clustered graph C in the c-planarity of C. Even if not explicitly stated, Theorem 10.1 has been already used in [CDPP05a].

Theorem 10.1 *An embedded flat clustered graph $C(G,T)$ is c-planar if and only if: (1) G is planar; (2) there exists a face f in G such that when f is chosen as outer face for G no cycle composed by vertices of the same cluster encloses a vertex of a different cluster; (3) it is possible to augment G to a graph G' by adding a subset of the candidate saturating edges of C so that no two added edges have a conflict and so that clustered graph $C'(G',T)$ is c-connected.*

Proof: First, we prove the necessity. The necessity of Condition 1 is trivial.

The necessity of Condition 2 easily descends from the definition of c-planarity. Namely, suppose that any plane embedding of G contains a cycle $\mathcal{C}$ composed by vertices belonging to cluster μ_i, such that $\mathcal{C}$ encloses a vertex v not belonging to μ_i. By definition of c-planar drawing, the region $R(\mu_i)$ representing μ_i in any drawing $\Gamma(C)$ of C contains $\mathcal{C}$, and hence either $R(\mu_i)$ is not simple, or it contains v, that does not belong to μ_i. By definition of c-planar drawing, in both cases $\Gamma(C)$ is not c-planar.

To prove that Condition 3 is necessary for the c-planarity of C, consider any c-planar drawing $\Gamma(C)$ of C. We show that it is possible to draw candidate saturating edges augmenting G to a graph G' so that the subgraph induced by each cluster in G' is connected and so that the augmented drawing $\Gamma'(C)$ is still c-planar.

Consider the region $R(\mu_i)$ representing in $\Gamma(C)$ a cluster μ_i. Subdivide $R(\mu_i)$ into connected open regions $R_j(\mu_i)$ delimited by the border of $R(\mu_i)$ and by the edges of G. Consider any region $R_j(\mu_i)$ that has on its border vertices of more than one connected component of $G(\mu_i)$. Edges connecting

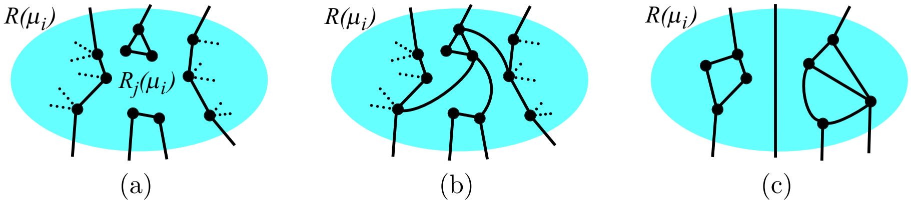


Figure 10.3: Illustrations for the proof of Theorem 10.1

vertices of different connected components can be drawn inside $R_j(\mu_i)$ so that the planarity of the drawing of G is maintained and so that the connected components of $G(\mu_i)$ appearing on the border of $R_j(\mu_i)$ form a unique connected component (see Figs. 10.3 (a) and 10.3 (b)). Notice that added edges are candidate saturating edges of C. After this step is repeated for every $R_j(\mu_i)$ all the connected components of $G(\mu_i)$ form a unique connected component. In fact, having two connected components in $\Gamma'(C)$ would imply that there is an edge-region crossing in $\Gamma(C)$ (see Fig. 10.3 (c)). After the augmentation is performed for every cluster μ_i the set of edges added to G satisfies the properties of Condition 3. Namely, no two added edges have a conflict since edges added to connect $G(\mu_i)$ and $G(\mu_j)$ for different clusters μ_i and μ_j are drawn inside non-overlapping regions $R(\mu_i)$ and $R(\mu_j)$.

Now we prove the sufficiency of Conditions 1, 2, and 3 for the c-planarity of C. Consider any planar drawing Γ of G in which no cycle composed by vertices of the same cluster encloses a vertex of a different cluster (such a drawing exists by Conditions 1 and 2). Consider a set S of candidate saturating edges of C satisfying Condition 3. Insert each edge e of S in Γ inside the face of G for which e is a candidate saturating edge. Since no two edges of S conflict each other, it is possible to do such an insertion so that the resulting drawing Γ' of the augmented graph G' is planar.

As long as G' has at least one edge e^* of S belonging to a cycle in which all the vertices are in the same cluster, remove e^* from G' and from Γ'. Clearly, such a removal leaves each cluster connected in G'. Moreover, after all such deletions no edge of any cycle in which all the vertices are in the same cluster belongs to S. Observe that removing such edges is not strictly necessary, since their presence does not alter the c-planarity of C'. However, their removal makes simpler the following construction steps.

For any cluster μ draw a region $R(\mu)$ representing μ in Γ' as a simple closed connected region surrounding the entire drawing of $G'(\mu)$. The border of $R(\mu)$

can be drawn so close to the border of the outer face of $G'(\mu)$ that (i) $R(\mu)$ does not enclose vertices that are outside the outer face of $G'(\mu)$, (ii) the border of $R(\mu)$ does not touch edges that are not incident to vertices of the outer face of $G'(\mu)$, and (iii) the borders of any two clusters do not intersect.

We prove that the resulting clustered drawing $\Gamma(C)$ of C is *c*-planar. By Condition 1, the drawing of G is planar. By construction, for each cluster μ, region $R(\mu)$ contains the drawing of $G'(\mu)$ in its interior. Suppose that a region $R(\mu)$ encloses a vertex $v \in V(\nu)$, with $\mu \neq \nu$. By the construction of region $R(\mu)$, this implies that there exists a cycle in $G'(\mu)$ enclosing v. However, since every cycle of G' in which all the vertices are in the same cluster is also a cycle of G, this would imply that Condition 2 is not satisfied by C. By the construction of regions $R(\mu)$ no two borders of different clusters intersect in $\Gamma(C)$. Finally, an edge-region crossing would imply an edge crossing in G', that is planar by Condition 3 and by the definition of saturator. □

Hence, given an embedded flat clustered graph $C(G,T)$, if Conditions 1 and 2 are satisfied by G, the problem of testing the *c*-planarity of C can be restated as the problem of testing if it is possible to select from multigraphs $\mathcal{G}_i$ a set of candidate saturating edges to enforce Condition 3 of Theorem 10.1. If such a set exists, we call it a *saturator* of C.

Lemma 10.1 *An embedded flat clustered graph $C(G,T)$ admits a saturator if and only if it admits an acyclic saturator.*

Proof: Consider any saturator S of C and denote by G' the embedded graph obtained by inserting each edge e of S inside the face of G for which e is a candidate saturating edge. As long as G' has at least one edge e^* of S belonging to a cycle in which all the vertices are in the same cluster, remove e^* from G'. After the removal the edges added to G are still a saturator of C, since, for each cluster μ, $G'(\mu)$ is connected and since the *c*-planarity of $C(G' \setminus e^*, T)$ is a consequence of the *c*-planarity of $C(G', T)$. Finally, observe that after all such deletions are performed no cycle composed of edges all belonging to S exists in G'. □

Hence, the problem of testing if an embedded flat clustered graph satisfying Conditions 1 and 2 of Theorem 10.1 is *c*-planar is reduced to the one of testing if there exists a spanning tree of each $\mathcal{G}_i$ such that no two edges in different spanning trees have a conflict.

10.3 A Characterization

In this section we study the c-planarity of those embedded flat clustered graphs in which each candidate saturating edge has a conflict with at most one other candidate saturating edge. We call an embedded flat clustered graph satisfying such a property to be *single-conflict*. The clustered graph of Fig. 10.4 is single-conflict, while the one of Fig. 10.2 is not.

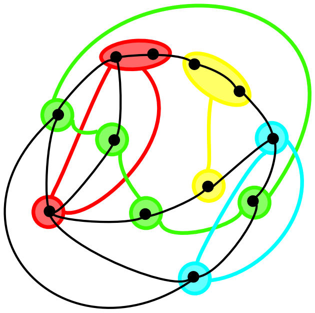


Figure 10.4: A single-conflict flat embedded clustered graph.

Consider a single-conflict embedded flat clustered graph $C(G,T)$ and, for any cluster μ_i in T, consider multigraph $\mathcal{G}_i$. We have the following structural lemma, showing that if two edges $e_1 = (u,v)$ and $e_2 = (x,w)$ of $\mathcal{G}_i$ cross, that is, vertices u, x, v, and w appear in this order on the border of the face f for which e_1 and e_2 are candidate saturating edges, then none of e_1 and e_2 can possibly cross an edge e_3 of a multigraph $\mathcal{G}_j$, with $i \neq j$.

Lemma 10.2 *If a graph $\mathcal{G}_i$ contains two crossing edges e_1 and e_2, then e_1 and e_2 have no conflict with edges of other multigraphs.*

Proof: Suppose, for a contradiction, that (i) C is a single-conflict embedded flat clustered graph, (ii) e_1 and e_2 are edges of $\mathcal{G}_i$, that is, e_1 and e_2 are candidate saturating edges for a cluster μ_i, (iii) e_1 and e_2 cross inside a face f of G, and (iv) e_1 has a conflict with an edge e_3 of a multigraph $\mathcal{G}_j$, with $j \neq i$, inside f (see Fig 10.5 (a)).

We claim that e_3 has conflicts with at least two edges of $\mathcal{G}_i$ and hence C is not a single-conflict embedded clustered graph. Let u and v, w and x, and y and z be the end-vertices of e_1, e_2, and e_3, respectively. If e_3 crosses e_2, the statement follows. Otherwise we can suppose without loss of generality, up

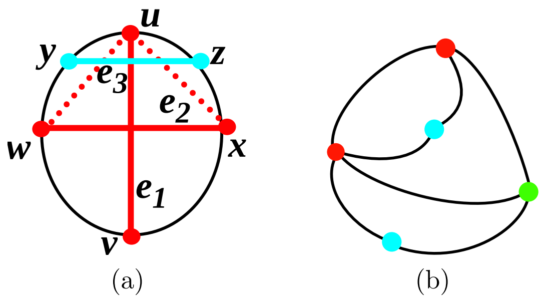


Figure 10.5: (a) Illustration for the proof of Lemma 10.2. (b) Illustration for the proof of Lemma 10.3. Graph $\mathcal{G}_i$ for the cyan cluster is not connected and there is no way of adding edges to the clustered graph to make the cyan cluster connected.

to a renaming of the vertices, that w, y, u, z, x, and v appear in this order around f. If vertices u and w do not belong to the same connected component of $G(\mu_i)$, then there exists in $\mathcal{G}_i$ an edge joining u and w that has a conflict with e_3 and the statement follows. Analogously, if vertices u and x do not belong to the same connected component of $G(\mu_i)$, then there exists in $\mathcal{G}_i$ an edge joining u and x that has a conflict with e_3 and the statement follows. However, either u and w, or u and x belong to different connected components of $G(\mu_i)$, otherwise u, w, and x would be in the same connected component of $G(\mu_i)$ and e_2 would not be a candidate saturating edge. □

By Lemma 10.3, we can assume that in the interesting cases the $\mathcal{G}_i$'s are connected (see Fig. 10.5 (b)).

Lemma 10.3 *If there exists a multigraph $\mathcal{G}_i$ that is not connected, then C is not c-planar.*

Proof: If a multigraph $\mathcal{G}_i$ is not connected, then adding to G any set of candidate saturating edges would not connect $G(\mu_i)$. Hence, by Theorem 10.1, C is not c-planar. □

There are edges in the $\mathcal{G}_i$'s that must be used in any saturator of C. Conversely, there are edges that will never be used in any saturator. Further, there are edges that can be supposed to belong to a saturator without altering the possibility to have one. Roughly speaking, such edges do not belong to the "core" of the problem. Hence, in the following we simplify the $\mathcal{G}_i$'s with an algorithm that either returns that C is not c-planar or returns a structure where there are no trivial choices. For this purpose, we define two operations on $\mathcal{G}_i$, that remove or collapse edges, to be used in the *simplification phase*.

The operation of *removing* an edge e from $\mathcal{G}_i$, corresponds to the choice of not using e as an edge of the saturator of C. Notice that, when an edge e is removed from $\mathcal{G}_i$, an edge of $\mathcal{G}_j$, with $i \neq j$, that eventually had a conflict with e does not have a conflict any longer.

The operation of *collapsing* an edge e with end-vertices u and v in $\mathcal{G}_i$ corresponds to the choice of using e as an edge of the saturator of C. It consists of (see Fig. 10.6): (i) deleting vertices u and v, (ii) removing from $\mathcal{G}_i$ all the edges between u and v, and (iii) inserting in $\mathcal{G}_i$ a new vertex w whose incident edges are those of u and v. The embedding of $\mathcal{G}_i$ is preserved. The collapsing operation "preserves" the conflicts. Namely, let e_i be an edge of $\mathcal{G}_i$ incident to u or to v but not to both. Suppose that e_i has a conflict (has not a conflict) with an edge e_j of $\mathcal{G}_j$, with $i \neq j$. After collapsing edge e in a new vertex w the edge incident to w corresponding to e_i has a conflict (resp. has not a conflict) with e_j. When an edge e is collapsed, the edge that conflicts with e, if any, is removed. In fact, collapsing e corresponds to choosing it in a saturator, hence no edge conflicting with e can be introduced in the saturator.

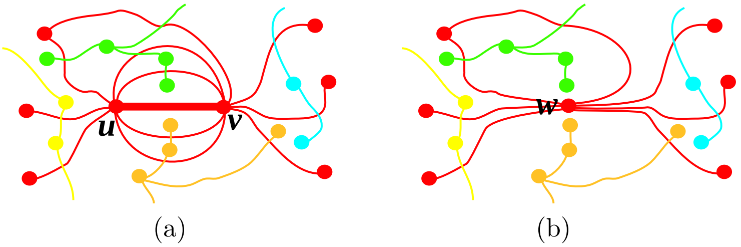


Figure 10.6: The operation of collapsing an edge (u, v): (a) Before collapsing (u, v). (b) After collapsing (u, v).

The simplification phase is as follows. Repeatedly modify the $\mathcal{G}_i$'s by applying one of the following simplifications. From now on, $\mathcal{G}_i$ denotes the multigraph obtained from the starting $\mathcal{G}_i$ after some simplifications have been performed.

Simplification 1: If there exists an edge e of a multigraph $\mathcal{G}_i$ that has no conflict, then collapse e in $\mathcal{G}_i$.

Simplification 2: If there exist a separating edge e_i and a non-separating edge e_j that are in multigraphs $\mathcal{G}_i$ and $\mathcal{G}_j$, respectively, and that conflict each other, then collapse e_i in $\mathcal{G}_i$ and remove e_j from $\mathcal{G}_j$.

Simplification 3: If there exist two separating edges e_i and e_j that are in multigraphs $\mathcal{G}_i$ and $\mathcal{G}_j$, respectively, and that conflict each other, then stop because C is not c-planar.

If the algorithm does not stop for non-c-planarity, we call the final multigraph $\mathcal{G}_i$ *candidate saturating graph* for cluster μ_i and we denote it by $\mathcal{G}_i^*$. Also, we say that μ_i *admits a candidate saturating graph*.

Observe that the above operations modify graphs $\mathcal{G}_i$. However, at any step of the simplification phase each edge e of $\mathcal{G}_i$ is associated with two vertices u and v and a face f of G meaning that if e is chosen to be in a saturator an edge between u and v is inserted in f. We preprocess $\mathcal{G}_i$ labeling each edge with its endpoints and with a face. Such labels are never changed by the operations described below. In the following, each time we add an edge e of $\mathcal{G}_i$ to a saturator, we add to G an edge between the endpoints and within the face specified by the label of e.

The following properties hold.

Property 10.1 *None of Simplifications 1, 2, and 3 could disconnect any multigraph $\mathcal{G}_i$.*

Proof: Simplification 1 collapses an edge of a multigraph $\mathcal{G}_i$. If $\mathcal{G}_i$ was connected before such a simplification, then it is still connected after that. Further, no edges of other multigraphs are removed when applying Simplification 1. Simplification 2 collapses an edge e_i of a multigraph $\mathcal{G}_i$ and removes the edge e_j of a multigraph $\mathcal{G}_j$ that had a conflict with e_i. However, if $\mathcal{G}_i$ was connected before such a simplification, then it is still connected after that, and since e_j is not a separating edge, then $\mathcal{G}_j$ remains connected after Simplification 2. Simplification 3 does not modify and hence does not disconnect any multigraph $\mathcal{G}_i$. □

Property 10.2 *None of Simplifications 1, 2, and 3 can create a self-loop in any multigraph $\mathcal{G}_i$.*

Proof: A self-loop in a multigraph $\mathcal{G}_i$ cannot be created by a removing operation. Further, when an edge e of a multigraph $\mathcal{G}_i$ is collapsed in a vertex w, edges parallel to e are removed. Hence, no self-loop is inserted in $\mathcal{G}_i$. □

Property 10.3 *The subgraphs induced by the collapsed edges are acyclic.*

Proof: Suppose that the subgraph induced by the set E of collapsed edges contains a cycle $\mathcal{C}$. Consider the last simplification s_m of the simplification phase that collapses one of the edges of $\mathcal{C}$, say edge $e = (u, v)$. A path $\mathcal{P}$ connecting u and v exists in E composed of candidate saturating edges that have been collapsed before s_m. The edges of $\mathcal{P}$ are collapsed in a single vertex w at the beginning of step s_m. By Property 10.2, vertex w has no self-loops, hence no edge connecting two vertices of $\mathcal{P}$ exists at step s_m. □

Property 10.4 *Candidate saturating graphs are planar embedded and edge* 2-*con-*
nected.

Proof: Each multigraph $\mathcal{G}_i$ before the simplification phase is planar embedded and the operations of removing and collapsing edges of $\mathcal{G}_i$ leave $\mathcal{G}_i$ planar embedded. By Property 10.1, multigraph $\mathcal{G}_i^*$ is connected. Further, if it has a separating edge e, then either e would be chosen to be in a saturator by one of Simplifications 1 and 2 (depending on whether e has no conflict or has a conflict with a non-separating edge) or C would not admit candidate saturating graphs (if e has a conflict with a separating edge). □

Property 10.5 *Any edge of a candidate saturating graph has exactly one conflict with an edge of a different candidate saturating graph.*

Proof: Any edge of a candidate saturating graph has at most one conflict with an edge of a different candidate saturating graph, since the embedded flat clustered graph is assumed to be single-conflict and operations of removing and collapsing edges do not introduce new conflicts. Any edge of a candidate saturating graph has at least one conflict with an edge of a different candidate saturating graph, otherwise it would be chosen to be in a saturator by Simplification 1. □

We now give lemmas in order to prove that each simplification performed by the algorithm preserves the possibility of finding a saturator of C. Consider simplification s_m, that is performed at a certain step of the simplification phase. Simplification s_m can be one of Simplifications 1, 2, or 3. Denote by $s_0, s_1, \ldots, s_{m-1}$ the simplifications that have been performed before s_m. Denote also by E the set of edges that have been collapsed while applying $s_0, s_1, \ldots, s_{m-1}$. Inductively, suppose that if an acyclic saturator of C exists, then there exists an acyclic saturator composed only of the edges of E plus some of the edges remaining in the $\mathcal{G}_i$'s after simplifications $s_0, s_1, \ldots, s_{m-1}$. This is indeed the case when no simplification has been performed yet.

Lemma 10.4 *Consider an edge e of $\mathcal{G}_i$ with no conflict. We have that C admits a saturator only if it admits an acyclic saturator containing e and containing the edges of E.*

Proof: Suppose C admits a saturator. Then, by Lemma 10.1, it admits an acyclic saturator. Moreover, by inductive hypothesis, it admits an acyclic saturator S such that $E \subseteq S$. If S contains e the statement follows. Otherwise, observe that since S is a saturator, there exists a set $S' \subseteq S$ of edges forming a path between the end-vertices u and v of e. Hence, the edges of $S' \cup \{e\}$ form a cycle. Notice that not all the edges of S' belong to E, otherwise u and v would not have been distinct vertices in $\mathcal{G}_i$ after simplifications $s_0, s_1, \ldots, s_{m-1}$. Hence, the set S^* of edges obtained from S by inserting e and by removing any edge of S' not in E is an acyclic saturator of C containing E and e. Namely, all the connected components of C are connected by a path of edges in S^* and since e has no conflict and S is a saturator, no two edges in S^* have a conflict. □

Lemma 10.5 *Consider two edges e_i and e_j of two distinct multigraphs $\mathcal{G}_i$ for cluster μ_i and $\mathcal{G}_j$ for cluster μ_j, respectively. Suppose that e_i and e_j conflict each other. Also, suppose that e_i is a separating edge, while e_j is not. Then C admits a saturator only if it admits an acyclic saturator containing e_i, containing E, and not containing e_j.*

Proof: Suppose C admits a saturator. Then, by Lemma 10.1, it admits an acyclic saturator. Moreover, by inductive hypothesis, it admits an acyclic saturator S such that $E \subseteq S$. Since at step s_m end-vertices u and v of e_i are in $\mathcal{G}_i$, no path composed by edges of E connects u and v. Moreover, since e_i is a separating edge, if e_i is not in S adding the edges of S to G would not connect $G(\mu_i)$. Hence $e_i \in S$. Since no two conflicting edges can be simultaneously in S, $e_j \notin S$. □

Lemma 10.6 *Consider two separating edges e_i and e_j of two distinct multigraphs $\mathcal{G}_i$ for cluster μ_i and $\mathcal{G}_j$ for cluster μ_j, respectively. Suppose that e_i and e_j conflict each other. We have that C is not c-planar.*

Proof: Suppose, for a contradiction, that C admits a saturator. Then, by inductive hypothesis, it admits an acyclic saturator S such that $E \subseteq S$. Since at step s_m the end-vertices u and v of e_i (the end-vertices w and x of e_j) are in $\mathcal{G}_i$ (are in $\mathcal{G}_j$), no path composed by edges of E connects u and v (connects w and x). Moreover, since e_i and e_j are separating edges, if e_i (e_j) is not in S, adding the edges of S to G would not connect $G(\mu_i)$ ($G(\mu_j)$). However, S cannot contain both e_i and e_j, that conflict each other. □

Let μ_i and μ_j be two distinct clusters admitting candidate saturating graphs $\mathcal{G}^*_i$ and $\mathcal{G}^*_j$, respectively. We define graph $\mathcal{G}^*_{i,j}$ as the planar embedded subgraph of $\mathcal{G}^*_i$ induced by the edges having a conflict with the edges of $\mathcal{G}^*_j$. We have (see Fig. 10.7):

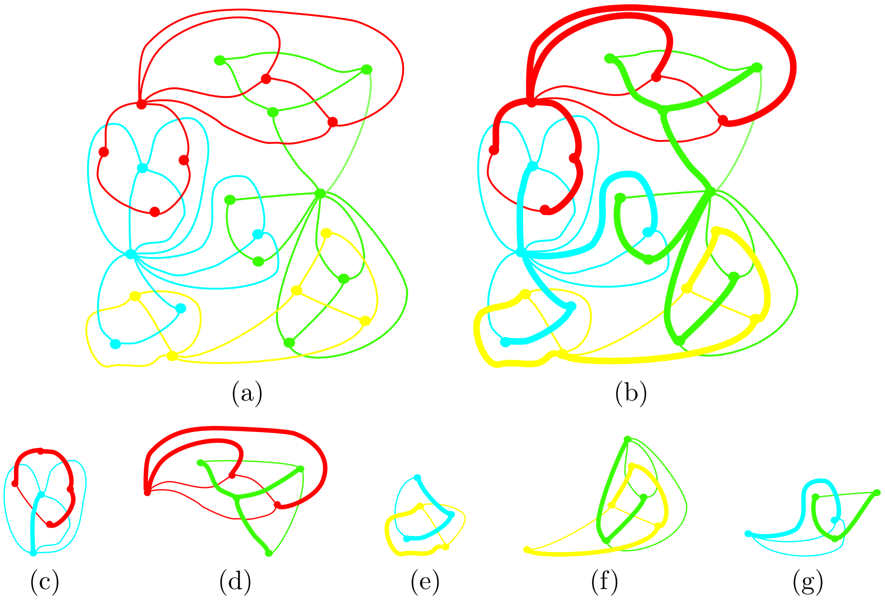


Figure 10.7: Illustrations for the statement of Theorem 10.2. (a) A set of candidate saturating graphs $\mathcal{G}^*_i$ for a single-conflict embedded flat clustered graph C. (b) A saturator of C. (c–g) Each picture contains graphs $\mathcal{G}^*_{i,j}$, $\mathcal{G}^*_{j,i}$, and spanning trees $\mathcal{T}^*_{i,j}$, $\mathcal{T}^*_{j,i}$ (in bold).

Theorem 10.2 *A single-conflict embedded flat clustered graph $C(G,T)$ is c-planar iff:*

1. *G is planar;*

2. *There exists a face f in G such that when f is chosen as outer face for G no cycle composed by vertices of the same cluster encloses a vertex of a different cluster;*

3. *Each cluster of C admits a candidate saturating graph;*

4. *For each pair of distinct clusters μ_i and μ_j, $\mathcal{G}^*_{i,j}$ is edge 2-connected; and*

5. *For each pair of distinct clusters μ_i and μ_j, $\mathcal{G}^*_{i,j}$ is dual to $\mathcal{G}^*_{j,i}$.*

Proof: First, we remark that each vertex of $\mathcal{G}^*_i$ corresponds to a distinct connected component of $G(\mu_i)$ after the edges chosen during the simplification phase have been added into the corresponding faces of G and that an edge connecting vertices u and v of $\mathcal{G}^*_i$ corresponds to an edge connecting a vertex of the connected component corresponding to u to a vertex of the connected component corresponding to v inside a face of G. Since the simplification phase preserves the possibility of finding an acyclic saturator S, then S can be found only if a set of edges can be selected from graphs $\mathcal{G}^*_i$ so that, after the edges of S are inserted into the faces of G, all the clusters induce connected graphs, no cycle composed of vertices of the same cluster has been created, and no two edges intersect. It follows that, in order to obtain an acyclic saturator S of C, a spanning tree of each $\mathcal{G}^*_i$ has to be selected such that no two edges of spanning trees of different graphs $\mathcal{G}^*_i$ and $\mathcal{G}^*_j$ have a conflict.

Let S be any acyclic saturator of C and let u and v be any two distinct vertices of any candidate saturating graph $\mathcal{G}^*_i$. We denote by $S(u,v)$ the unique path connecting u and v in the spanning tree of $\mathcal{G}^*_i$ contained in S. We remark that such a path exists, otherwise cluster μ_i would not induce a connected graph after adding the edges of S to G, and is unique, otherwise the chosen saturator S would not be acyclic. If edges e_i and e_j of different candidate saturating graphs $\mathcal{G}^*_i$ and $\mathcal{G}^*_j$ conflict each other, we write $e_i \oplus e_j$.

The necessity of Conditions 1 and 2 descends from the necessity of Conditions 1 and 2 of Theorem 10.1. We prove the necessity of Condition 3. Suppose that C does not admit candidate saturating graphs. Two cases are possible: Either before the simplification phase one of the $\mathcal{G}_i$'s is not connected, or during the simplification phase two separating conflicting edges are found. In the former case the non-c-planarity of C descends from Lemma 10.3, in the latter case from Lemma 10.6.

Now we deal with Condition 4. Suppose that $\mathcal{G}^*_{i,j}$ is not connected and denote by u_1 and u_2 vertices in different connected components. Suppose, for a contradiction, that an acyclic saturator S of C exists. Consider path $S(u_1, u_2)$ (see Fig. 10.9 (a)). Since u_1 and u_2 are in different connected components of $\mathcal{G}^*_{i,j}$, there exists an edge $(u_3, u_4) \in S(u_1, u_2)$ such that $(u_3, u_4) \oplus (w_1, w_2)$, where $(w_1, w_2) \in \mathcal{G}^*_k$, with $k \neq i, j$. Consider path $S(w_1, w_2)$. Each edge of $S(w_1, w_2)$ cannot have a conflict with any edge of $S(u_1, u_2)$, otherwise S would contain two conflicting edges, and neither can it have a conflict with any edge (v_1, v_2) of $\mathcal{G}^*_{j,i}$, otherwise (v_1, v_2) would conflict with two candidate saturating edges. Hence, $\mathcal{G}^*_{j,i}$ has at least two connected components. Let v_3 and v_4 be two vertices in such components, respectively. Then, $S(v_3, v_4)$ either contains an edge (v_5, v_6) such that $(v_5, v_6) \oplus (w_3, w_4)$, with $(w_3, w_4) \in S(w_1, w_2)$, implying that S contains two conflicting edges, or contains an edge (v_5, v_6) conflicting with edge (w_1, w_2), implying that (w_1, w_2) conflicts with two candidate saturating edges.

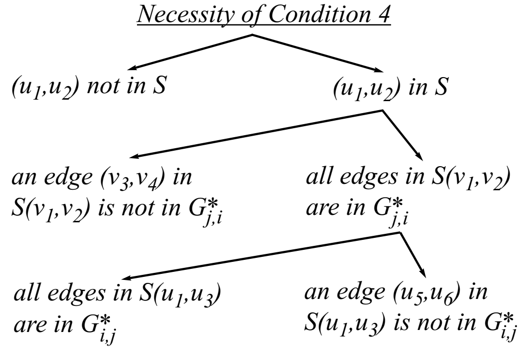

Necessity of Condition 4
(u_1,u_2) not in S
(u_1,u_2) in S
an edge (v_3,v_4) in $S(v_1,v_2)$ is not in $G^*_{j,i}$
all edges in $S(v_1,v_2)$ are in $G^*_{j,i}$
all edges in $S(u_1,u_3)$ are in $G^*_{i,j}$
an edge (u_5,u_6) in $S(u_1,u_3)$ is not in $G^*_{i,j}$

Figure 10.8: Proof of the necessity of Condition 4. Edge (u_1, u_2) is a separating edge that has a conflict with an edge (v_1, v_2). If $(u_1, u_2) \in S$ and all the edges of $S(v_1, v_2)$ belong to $\mathcal{G}^*_{j,i}$, then (u_3, u_4) is an edge that has a conflict with an edge of $S(v_1, v_2)$. Vertices u_1 and u_3 are both internal to cycle $S(v_1, v_2) \cup (v_1, v_2)$.

Now suppose that $\mathcal{G}^*_{i,j}$ has a separating edge (u_1, u_2). By construction $(u_1, u_2) \oplus (v_1, v_2)$, where $(v_1, v_2) \in \mathcal{G}^*_{j,i}$. Suppose, for a contradiction, that a saturator S of C exists. Fig. 10.8 shows the strategy of the proof of such a contradiction.

- If $(u_1, u_2) \notin S$, then consider $S(u_1, u_2)$ (see Fig. 10.9 (b)). Since (u_1, u_2) is a separating edge for $\mathcal{G}^*_{i,j}$, there exists an edge $(u_3, u_4) \in S(u_1, u_2)$ such that $(u_3, u_4) \oplus (w_1, w_2)$, where $(w_1, w_2) \in \mathcal{G}^*_k$, with $k \neq i, j$. Hence, $S(w_1, w_2)$ either contains an edge (w_3, w_4) such that $(w_3, w_4) \oplus (u_5, u_6)$, with $(u_5, u_6) \in S(u_1, u_2)$, implying that S contains two conflicting edges, or contains an edge (w_3, w_4) conflicting with (u_1, u_2), implying that (u_1, u_2) conflicts with two candidate saturating edges.

- If $(u_1, u_2) \in S$, then consider $S(v_1, v_2)$.
  - If an edge $(v_3, v_4) \in S(v_1, v_2)$ is such that $(v_3, v_4) \oplus (w_1, w_2)$, where $(w_1, w_2) \in \mathcal{G}^*_k$, with $k \neq i, j$, a contradiction is obtained as in the previous case (see Fig. 10.9 (c)).
  - Otherwise, all the edges of $S(v_1, v_2)$ belong to $\mathcal{G}^*_{j,i}$. Consider any edge $(v_3, v_4) \in S(v_1, v_2)$ and edge $(u_3, u_4) \in \mathcal{G}^*_{i,j}$ such that $(u_3, u_4) \oplus (v_3, v_4)$. Let u_1 (u_3) be the endpoint of (u_1, u_2) (resp. of (u_3, u_4)) inside cycle $S(v_1, v_2) \cup (v_1, v_2)$.
    - * If $u_1 = u_3$ or if all the edges of $S(u_1, u_3)$ have conflicts with edges of $\mathcal{G}^*_{j,i}$ (see Fig. 10.9 (d)), consider path $S(u_2, u_4)$. Then there exists an edge $(u_5, u_6) \in S(u_2, u_4)$ such that $(u_5, u_6) \oplus (w_1, w_2)$, where $(w_1, w_2) \in \mathcal{G}^*_k$, with $k \neq i, j$, otherwise (u_1, u_2) would not be a separating edge. Hence, $S(w_1, w_2)$ either contains an edge (w_3, w_4) such that $(w_3, w_4) \oplus (u_7, u_8)$, with $(u_7, u_8) \in S(u_3, u_4)$, implying that S contains two conflicting edges, or an edge (w_3, w_4) such that $(w_3, w_4) \oplus (u_3, u_4)$ implying that (u_3, u_4) conflicts with two candidate saturating edges.
    - * If $u_1 \neq u_3$ and $S(u_1, u_3)$ contains at least one edge (u_5, u_6) such that $(u_5, u_6) \oplus (w_1, w_2)$, where $(w_1, w_2) \in \mathcal{G}^*_k$, with $k \neq i, j$ (see Fig. 10.9 (e)), then $S(w_1, w_2)$ contains: (i) either an edge (w_3, w_4) such that $(w_3, w_4) \oplus (v_5, v_6)$, with $(v_5, v_6) \in S(v_1, v_2)$, implying that S contains two conflicting edges, (ii) or an edge (w_3, w_4) such that $(w_3, w_4) \oplus (u_7, u_8)$, with $(u_7, u_8) \in S(u_2, u_3)$ implying that S contains two conflicting edges, (iii) or an edge (w_3, w_4) such that $(w_3, w_4) \oplus (u_3, u_4)$, implying that (u_3, u_4) conflicts with two candidate saturating edges, (iv) or an edge (w_3, w_4) such that $(w_3, w_4) \oplus (v_1, v_2)$, implying that (v_1, v_2) conflicts with two candidate saturating edges.

Now we prove the necessity of Condition 5. By definition, each edge of $\mathcal{G}^*_{i,j}$ has a conflict with (and hence is dual to) one edge of $\mathcal{G}^*_{j,i}$ and vice versa.

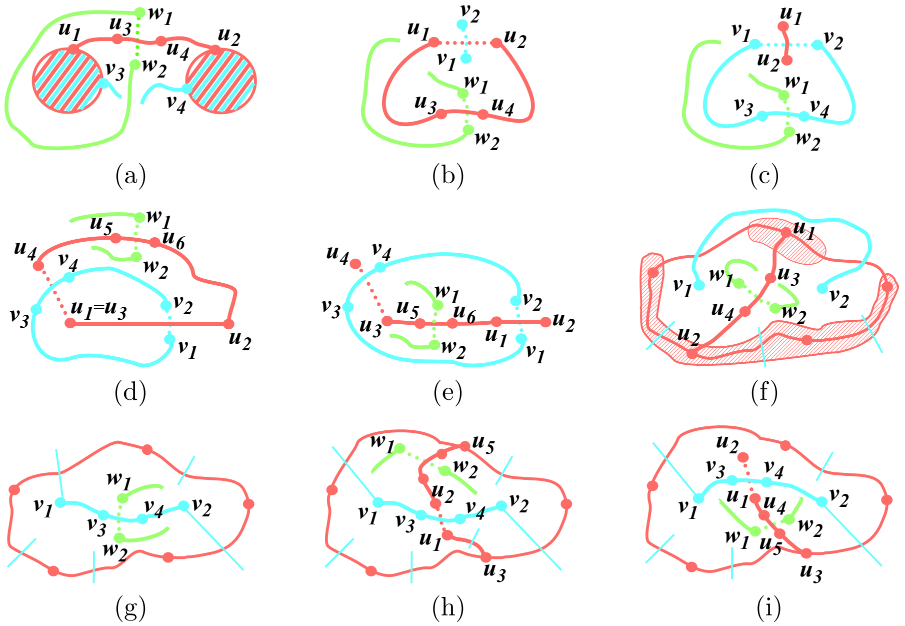


Figure 10.9: Illustrations for the necessity of the conditions of Theorem 10.2. Edges of $\mathcal{G}_i^*$ are red, edges of $\mathcal{G}_j^*$ are light blue, and edges of $\mathcal{G}_k^*$ are green.

Moreover, by the necessity of Condition 4, we can assume that both $\mathcal{G}_{i,j}^*$ and $\mathcal{G}_{j,i}^*$ are edge 2-connected. Hence $\mathcal{G}_{i,j}^*$ is not dual to $\mathcal{G}_{j,i}^*$ only if there is a face of $\mathcal{G}_{i,j}^*$ that contains in its interior two vertices of $\mathcal{G}_{j,i}^*$, or vice versa. Suppose w.l.o.g. that a face f of $\mathcal{G}_{i,j}^*$ contains in its interior two vertices v_1 and v_2 of $\mathcal{G}_{j,i}^*$. Suppose, for a contradiction, that a saturator S of C exists. Consider path $S(v_1, v_2)$. Fig. 10.10 shows the strategy of the proof of such a contradiction.

- If $S(v_1, v_2)$ is composed in part by vertices inside f and in part by vertices outside f (see Fig. 10.9 (f)), consider two vertices u_1 and u_2 in different connected components, disconnected by $S(v_1, v_2)$, of f. Consider $S(u_1, u_2)$. There exists an edge $(u_3, u_4) \in S(u_1, u_2)$ such that $(u_3, u_4) \oplus (w_1, w_2)$, where $(w_1, w_2) \in \mathcal{G}_k^*$, with $k \neq i, j$, otherwise f would not be a face. Hence, $S(w_1, w_2)$ either contains an edge (w_3, w_4) such that $(w_3, w_4) \oplus (u_5, u_6)$, with $(u_5, u_6) \in S(u_1, u_2)$, implying that S contains two conflicting edges, or contains an edge (w_3, w_4) conflicting with

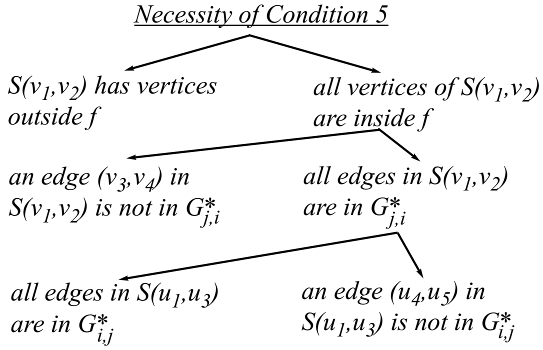

Necessity of Condition 5
S(v1,v2) has vertices outside f
all vertices of S(v1,v2) are inside f
an edge (v3,v4) in S(v1,v2) is not in G*j,i
all edges in S(v1,v2) are in G*j,i
all edges in S(u1,u3) are in G*i,j
an edge (u4,u5) in S(u1,u3) is not in G*i,j

Figure 10.10: Proof of the necessity of Condition 5. Vertices v_1 and v_2 are both in face f. If all the edges of $S(v_1, v_2)$ belong to $\mathcal{G}^*_{j,i}$, then (u_1, u_2) is an edge that has a conflict with an edge of $S(v_1, v_2)$. Vertex u_3 is in f.

an edge $(u_5, u_6) \in f$, implying that (u_5, u_6) conflicts with two candidate saturating edges.

- Otherwise, $S(v_1, v_2)$ is composed by vertices all lying inside f.
  - If there exists an edge $(v_3, v_4) \in S(v_1, v_2)$ such that $(v_3, v_4) \oplus (w_1, w_2)$, where $(w_1, w_2) \in \mathcal{G}^*_k$, with $k \neq i, j$ (see Fig. 10.9 (g)), then $S(w_1, w_2)$ contains: (i) either an edge (w_3, w_4) such that $(w_3, w_4) \oplus (v_5, v_6)$, with $(v_5, v_6) \in S(v_1, v_2)$, implying that S contains two conflicting edges, (ii) or an edge (w_3, w_4) such that $(w_3, w_4) \oplus (u_1, u_2)$, with $(u_1, u_2) \in f$, implying that (u_1, u_2) conflicts with two candidate saturating edges, (iii) or an edge (w_3, w_4) such that $(w_3, w_4) \oplus (v_5, v_6)$, with (v_5, v_6) dual to an edge of f, implying that (v_5, v_6) conflicts with two candidate saturating edges.
  - Otherwise, each edge of $S(v_1, v_2)$ is dual to an edge of $\mathcal{G}^*_{i,j}$. Consider any edge (u_1, u_2) dual to an edge of $S(v_1, v_2)$.
    - ∗ If $u_1 \in f$ or if there exists a vertex $u_3 \in f$ such that all the edges of $S(u_1, u_3)$ conflict with edges of $\mathcal{G}^*_{j,i}$ (see Fig. 10.9 (h)), then $u_2 \notin f$ and there exists no vertex u_4 in f such that all the edges of $S(u_2, u_4)$ conflict with edges of $\mathcal{G}^*_{j,i}$, otherwise f would not be a face. Consider any vertex $u_5 \in f$ and path $S(u_2, u_5)$. Then, there exists an edge in $S(u_2, u_5)$ that has a

conflict with an edge (w_1, w_2) in $\mathcal{G}_k^*$, with $k \neq i, j$. Hence, path $S(w_1, w_2)$ contains: (i) either an edge (w_3, w_4) such that $(w_3, w_4) \oplus (v_5, v_6)$, with $(v_5, v_6) \in S(v_1, v_2)$, implying that S contains two conflicting edges, (ii) or an edge (w_3, w_4) such that $(w_3, w_4) \oplus (u_6, u_7)$, with $(u_6, u_7) \in S(u_2, u_5)$, implying that S contains two conflicting edges, (iii) or an edge (w_3, w_4) such that $(w_3, w_4) \oplus (u_6, u_7)$, with $(u_6, u_7) \in f$, implying that (u_6, u_7) conflicts with two candidate saturating edges, (iv) or an edge (w_3, w_4) such that $(w_3, w_4) \oplus (v_5, v_6)$, with (v_5, v_6) dual to an edge in f, implying that (v_5, v_6) conflicts with two candidate saturating edges.

∗ If $u_1 \notin f$ and there exists no vertex $u_3 \in f$ such that every edge of $S(u_1, u_3)$ conflicts with an edge of $\mathcal{G}_{j,i}^*$ (see Fig. 10.9 (i)), then there exists a vertex $u_3 \in f$ such that $S(u_1, u_3)$ contains an edge (u_4, u_5) such that $(u_4, u_5) \oplus (w_1, w_2)$, with $(w_1, w_2) \in \mathcal{G}_k^*$, with $k \neq i, j$, and a contradiction is derived as in the previous case.

Now we prove the sufficiency of Conditions 1, 2, 3, 4, and 5, for the c-planarity of $C(G, T)$. Consider any planar drawing of G satisfying Conditions 1 and 2 and hence satisfying Conditions 1 and 2 of Theorem 10.1. We show how to construct an acyclic saturator S of C satisfying Condition 3 of Theorem 10.1. Apply the simplification phase. As a result, get an acyclic set E of edges already chosen to be in S and a candidate saturating graph $\mathcal{G}_i^*$ for each cluster μ_i (this can be done since C satisfies Condition 3).

Order the clusters in whichever way $\mu_1, \mu_2, \dots, \mu_m$. For any pair of clusters μ_i and μ_j, with $i < j$, choose a spanning tree $\mathcal{T}_{i,j}^*$ of $\mathcal{G}_{i,j}^*$ (a spanning tree of $\mathcal{G}_{i,j}^*$ can always be found since, by Condition 4, $\mathcal{G}_{i,j}^*$ is edge 2-connected). Remove from $\mathcal{G}_{j,i}^*$ all the edges dual to edges of $\mathcal{T}_{i,j}^*$, obtaining a graph $\mathcal{T}_{j,i}^*$. We claim that $\mathcal{T}_{j,i}^*$ is a spanning tree of $\mathcal{G}_{j,i}^*$. By Condition 5, $\mathcal{G}_{i,j}^*$ and $\mathcal{G}_{j,i}^*$ are dual graphs, and, since they are edge 2-connected, the edges of a cycle in $\mathcal{G}_{i,j}^*$ are dual to the edges of a cutset in $\mathcal{G}_{j,i}^*$, and vice versa (Lemma 1.4 of [NC88]). Hence, if $\mathcal{T}_{j,i}^*$ has more than one connected component, then the edges removed from $\mathcal{G}_{j,i}^*$ form a cutset for $\mathcal{G}_{j,i}^*$, and the edges of $\mathcal{T}_{i,j}^*$ form a cycle, contradicting the hypothesis that $\mathcal{T}_{i,j}^*$ is a tree. Moreover, if a set of edges of $\mathcal{T}_{j,i}^*$ is a cycle, then the edges dual to such a cycle form a cutset for $\mathcal{G}_{i,j}^*$, contradicting the hypothesis that $\mathcal{T}_{i,j}^*$ is spanning for $\mathcal{G}_{i,j}^*$.

For any pair of clusters μ_i and μ_j, with $i < j$, add the edges of $\mathcal{T}_{i,j}^*$ and the edges of $\mathcal{T}_{j,i}^*$ to S. We claim that S is an acyclic saturator of C. Namely, we prove that (1) no two edges of S have a conflict, (2) adding the edges of S to

G connects the subgraph induced by each cluster, and (3) adding the edges of S to G does not create cycles composed by vertices all belonging to the same cluster.

1. *No two edges of S have a conflict:* Edges chosen in the simplification phase do not conflict each other by construction. Such edges do not conflict with edges of trees $\mathcal{T}^*_{i,j}$. In fact, an edge in $\mathcal{T}^*_{i,j}$ conflicts only with an edge in $\mathcal{G}^*_j$, with $i \neq j$. By construction, edges of the $\mathcal{T}^*_{i,j}$'s do not conflict each other.

2. *Adding the edges of S to G connects the subgraph induced by each cluster:* Distinct connected components of $G(\mu_i)$ are represented after the simplification phase by distinct vertices in $\mathcal{G}^*_i$, that is edge 2-connected and that is partitioned in edge 2-connected subgraphs $\mathcal{G}^*_{i,j}$. Since a spanning tree is chosen to be in S for any $\mathcal{G}^*_{i,j}$, we have that $\bigcup_j \mathcal{T}^*_{i,j}$ is spanning for $\mathcal{G}^*_i$ and $G'(\mu_i)$ has exactly one connected component. Recall that $G'(\mu_i)$ is the graph obtained by adding the edges of the saturator to $G(\mu_i)$.

3. *Adding the edges of S to G does not create cycles composed by vertices all belonging to the same cluster:* Suppose that $G'(\mu_i)$ has a cycle $\mathcal{C}$ containing an edge of S. By construction, edges chosen in the simplification phase only join different connected components of $G(\mu_i)$ and no edge of $\mathcal{C}$ could belong to some $\mathcal{G}^*_{i,j}$, otherwise $G'(\mu_j)$ would be disconnected.

Since S is an acyclic saturator of C, the conditions of Theorem 10.1 are satisfied by C, that hence is c-planar. □

Theorem 10.3 *An embedded flat clustered graph $C(G, T)$ with at most five vertices per face is c-planar if and only if:*

1. *G is planar;*

2. *There exists a face f in G such that when f is chosen as outer face for G no cycle composed by vertices of the same cluster encloses a vertex of a different cluster;*

3. *Each cluster of C admits a candidate saturating graph;*

4. *For each pair of distinct clusters μ_i and μ_j, $\mathcal{G}^*_{i,j}$ is edge 2-connected; and*

5. *For each pair of distinct clusters μ_i and μ_j, $\mathcal{G}^*_{i,j}$ is dual to $\mathcal{G}^*_{j,i}$.*

Proof: Consider any face f of G. Since f has at most five vertices, it has at most two connected components of each cluster, so it has at most one candidate saturating edge for each cluster. Since at least two vertices are necessary for each candidate saturating edge, f contains candidate saturating edges for at most two clusters. Hence, C is a single-conflict embedded flat clustered graph and the statement follows from Theorem 10.2. □

10.4 An Efficient C-Planarity Testing Algorithm

In this section we use Theorem 10.3 to derive a linear time and space c-planarity testing algorithm for embedded flat clustered graphs with at most five vertices per face. The algorithm can be extended to test the c-planarity of single-conflict embedded flat clustered graphs relying on Theorem 10.2. However, it turns out that such an extension exploits several technicalities, in order to deal with a number of candidate saturating edges that can be asymptotically more than linear in the number of vertices of the clustered graph. Hence, to improve the readability of the section, we give the algorithm for the case of embedded flat clustered graphs with at most five vertices per face, while emphasizing the steps of the algorithm that have to be modified to deal with single-conflict embedded flat clustered graphs.

Let $C(G,T)$ be an n-vertex embedded flat clustered graph with at most five vertices per face. To test Condition 1 of Theorem 10.3, it is sufficient to test if G is a planar embedding. This can be done in $O(n)$ time and space with the techniques in [Kir88].

To test Condition 2, we observe that a face exists satisfying such a condition if and only if the embedded clustered graph is *hole-free*, that is, chosen an arbitrary face as external, there exists no cycle $\mathcal{C}$ that is composed by vertices of the same cluster μ and that separates two vertices both belonging to clusters different from μ (see Fig. 10.11).

A linear-time algorithm for checking if an embedded clustered graph is hole-free has been provided in [Dah98] in the case of c-connected clustered graphs. We can use the same algorithm because of the following lemma.

Lemma 10.7 *Let $C(G,T)$ be an embedded clustered graph. Let $C'(G,T')$ be the embedded c-connected clustered graph obtained from C as follows. Each node ν of T is replaced in T' by nodes $\nu_1,\dots,\nu_h$, one for each of the $h \geq 1$ connected components of $G(\nu)$. Let $\mu_1,\dots,\mu_k$ be the nodes replacing the parent*

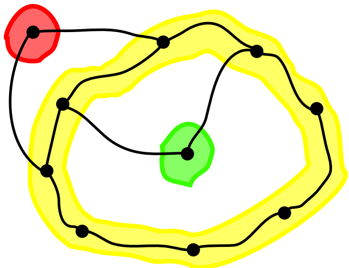


Figure 10.11: A hole in an embedded clustered graph. A hole consists of a cycle that is composed by vertices of the same cluster μ and that separates two vertices both belonging to clusters different from μ. An embedded clustered graph having no hole is said to be *hole-free*.

μ of ν. The parent of ν_j in T' is the node μ_i such that $G(\nu_j)$ is a subgraph of $G(\mu_i)$. We have that C is hole-free if and only if C' is hole-free.

Proof: Suppose that C is hole-free and suppose, for a contradiction, that C' is not hole-free. Choose arbitrarily in G an external face. Then, a cycle $\mathcal{C}$ of G exists composed by vertices of the same cluster $\mu_i \in T'$ such that $\mathcal{C}$ has a vertex v_1 inside and a vertex v_2 outside both belonging to clusters in T' different from μ_i. Consider cluster $\mu \in T$ that is replaced in T' by a set of clusters among which there is μ_i. By construction the vertices of $\mathcal{C}$ belong to μ in C.

We claim that there exists a vertex inside $\mathcal{C}$ that does not belong to μ. Since $v_1 \notin \mu_i$, there are two cases: Either $v_1 \notin \mu$, or $v_1 \in \mu$. In the first case the claim directly follows. In the second case, since v_1 and $\mathcal{C}$ belong to μ but are in different clusters in C', they belong to different connected components of $G(\mu)$. Consider any path internal to $\mathcal{C}$ connecting v_1 to a vertex of $\mathcal{C}$. Such a path exists, otherwise G would not be connected. The vertices of such a path cannot all belong to μ, otherwise v_1 and $\mathcal{C}$ would be in the same connected component of $G(\mu)$. Hence, there exists a vertex internal to $\mathcal{C}$ not belonging to μ and the claim follows. A similar argument proves that there exists a vertex outside $\mathcal{C}$ that does not belong to μ, that gives the desired contradiction.

Now suppose C' is hole-free and suppose, for a contradiction, that C is not hole-free. Choose arbitrarily in G an external face. Then, a cycle $\mathcal{C}$ of G exists composed by vertices of the same cluster $\mu \in T$ such that $\mathcal{C}$ has a vertex v_1 inside and a vertex v_2 outside both belonging to clusters in T different from

μ. Then, consider cluster μ_i containing $\mathcal{C}$ in C'. Since $v_1, v_2 \notin \mu$, we have that $v_1, v_2 \notin \mu_i$, that gives the desired contradiction. □

In order to test Condition 3 we need to create multigraphs $\mathcal{G}_i$. This is done in $O(n)$ time as follows.

- *Connected Components.* For each node μ of T compute the connected components of $G(\mu)$. This is easily done in linear time and space. Each vertex v of $G(\mu)$ is labelled by a name uniquely associated with the connected component of $G(\mu)$ vertex v belongs to.

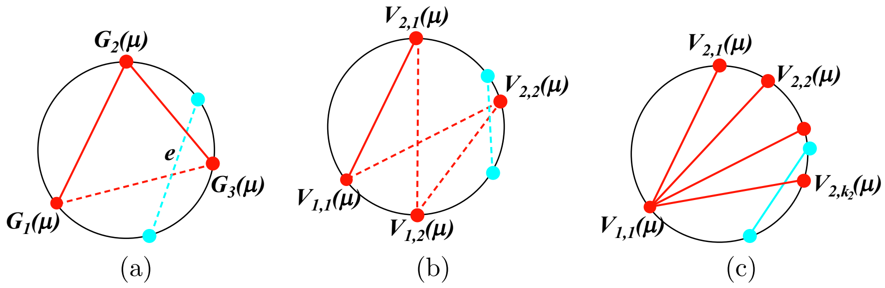
- *Candidate saturating edges.* We insert candidate saturating edges inside the faces of G. Consider a face f. Construct maximal sequences of vertices that are consecutive on the border of f and that belong to the same cluster. For any two sequences V_1 and V_2 that have vertices belonging to the same cluster μ, take a vertex $v_1 \in V_1$ and a vertex $v_2 \in V_2$. Test in constant time if the connected component $G_i(\mu)$ of $G(\mu)$ labelling v_1 is different from the connected component $G_j(\mu)$ labelling v_2. If $G_i(\mu)$ is not the same connected component of $G_j(\mu)$, then insert a candidate saturating edge between v_1 and v_2. As already discussed in the proof of Theorem 10.3, at most two edges are inserted inside f. Since f has at most five vertices, the described insertion can be performed in constant time and hence in linear time for all the faces of G.

  This step is more tricky when considering single-conflict clustered graphs, where faces can have a linear number of vertices. In that case, in order to achieve linear time special care must be taken when considering groups of candidate saturating edges between vertices of the same cluster and when determining the conflicts between candidate saturating edges.

  Namely, consider a face f and a cluster μ having connected components $G_1(\mu), \dots, G_k(\mu)$ in f.

  If $k = 1$ no candidate saturating edge is inserted in f.

  If $k > 2$ (see Fig. 10.12 (a)), then we insert in f one candidate saturating edge between any vertex of $G_i(\mu)$ and any vertex of $G_{i+1}(\mu)$, for $i = 1, \dots, k-1$. In fact, in this case, since C is single-conflict, none of such edges has a conflict with any other candidate saturating edge e (otherwise e would have more than one conflict). Hence, since such edges are conflict-free no other edge is required in order to connect the components of cluster μ in f.



(a) (b) (c)

Figure 10.12: Candidate saturating edges for a cluster μ_i in a single-conflict clustered graph: (a) $k > 2$. (b) $k = 2$, $k_1 > 1$, and $k_2 > 1$. (c) $k = 2$ and $k_1 = 1$. The candidate saturating edges that are added to the graphs $\mathcal{G}_i$'s are solid. Dashed cyan edges correspond to candidate saturating edges that cannot exist, otherwise the clustered graph would not be single-conflict. Dashed red edges correspond to candidate saturating edges that are not needed to connect the components of μ_i in f.

Suppose that $k = 2$ and let $V_{1,1}(\mu), \dots, V_{1,k_1}(\mu)$ ($V_{2,1}(\mu), \dots, V_{2,k_2}(\mu)$) be the maximal sequences of vertices that are consecutive on the border of f and that belong to $G_1(\mu)$ (resp. to $G_2(\mu)$).

If both $k_1 > 1$ and $k_2 > 1$ (see Fig. 10.12 (b)), we add a candidate saturating edge between any vertex of $V_{1,1}(\mu)$ and any vertex of $V_{2,1}(\mu)$. Such an edge is conflict-free and we can repeat the above arguments to show that no other edge is required to connect the components of μ in f.

If either $k_1 = 1$ or $k_2 = 1$, say $k_1 = 1$ (see Fig. 10.12 (c)), we add edges between any vertex of $V_{1,1}(\mu)$ and any vertex of $V_{2,i}(\mu)$ (with $i = 1, \dots, k_2$). Observe that such edges might have conflicts.

By the above discussion, the number of candidate saturating edges inserted for each cluster μ inside f is linear in the number of maximal sequences of vertices that are consecutive on the border of f and that belong to $G(\mu)$. It follows that a total linear number of candidate saturating edges are inserted into the faces of G. Further, such edges are sufficient to find a saturator for the clustered graph, if any such a saturator exists.

At this point we detect conflicts. We traverse the border of f in clockwise direction starting at any vertex. During the traversal, we maintain a list of encountered edges in a stack P. At each encountered vertex v we do what follows: We consider the candidate saturating edges incident to v in clockwise order; for each edge e, if e has never been encountered we insert e into P; otherwise, the first end-vertex of e has already been encountered and e is already in P. We check if e has a conflict with the top edge e' of P. If yes, we record the conflict and remove e and e' from P. If not, we remove e from P. Such a procedure detects all conflicts among candidate saturating edges. In fact, the conflict structure of the candidate saturating edges is parenthetic, due to the restriction to single-conflict clustered graphs.

- *Multigraphs* $\mathcal{G}_i$. Consider cluster μ_i. Add a vertex to $\mathcal{G}_i$ for each connected component of $G(\mu_i)$. For each of the above mentioned candidate saturating edges (u, v), insert an edge between the connected components of u and v. For each edge e in a multigraph $\mathcal{G}_i$, we record the edge e^* that has a conflict with e, if any. The construction of the $\mathcal{G}_i$'s can be done such that their embeddings are those induced by the adjacencies of the faces of G. Further, such a construction can be done in linear time and space because of the following:

  Property 10.6 $\sum_{\mu_i} |\mathcal{G}_i| = O(n)$, *where* $|\mathcal{G}_i|$ *is the size of the graph.*

  Proof: The total number of vertices of the $\mathcal{G}_i$'s is at most the number of vertices of G, hence it is bounded by n.

  If each face of G has at most five vertices the proof is trivial. In fact, there are at most two candidate saturating edges for each face. Hence, the total number of edges of the $\mathcal{G}_i$'s is $O(n)$.

  On the other hand, when considering single-conflict embedded flat clustered graphs, that can generally have faces with a linear number of incident vertices, we apply the algorithm described above, that inserts for each face f only a number of edges that is linear in the size of f. □

Now we show how to test if Condition 3 of Theorem 10.3 is satisfied. First, test if the $\mathcal{G}_i$'s are connected. If not, return non-c-planar.

We equip each $\mathcal{G}_i$ with a data structure supporting the following update operations, which are trivial graph operations and that can be hence be performed in constant time: Remove an edge, collapse (identify the end-vertices of) an edge and merge the embeddings of its end-vertices. Observe the difference between the above definition of the collapse operation and the one given in Sect. 10.3, where the edges between the end-vertices are removed.

Next, we show how to apply the simplification phase. We first deal with conflict-free edges, that are edges with no conflict, and we will apply Simplification 1 till the multigraphs $\mathcal{G}_i$'s have no conflict-free edge. Second, we will handle separating edges by either applying Simplification 2 till the multigraphs $\mathcal{G}_i$'s have no separating edge or the non-c-planarity of C has been established.

- *Conflict-free edges.* Extract from all $\mathcal{G}_i$'s the candidate saturating edges that have no conflict. Insert all such edges into a set called $\mathcal{F}$. For each edge e of $\mathcal{F}$ compute the set $\mathcal{E}$ of edges parallel to e. Such computations are easily performed in linear time.

  Construct the set $\mathcal{F}'$ of the edges of any spanning forest of $\mathcal{F}$. The edges of $\mathcal{F}'$ are collapsed as follows. Let $\mathcal{F}''$ be the set containing the edges that have no conflict after the edges of $\mathcal{F}'$ have been collapsed. We construct $\mathcal{F}''$ as follows. Initialize $\mathcal{F}'' = \emptyset$. Take each edge e_1 of $\mathcal{F}'$. Consider the set $\mathcal{E}$ of edges parallel to e_1. For each edge $e_2 \neq e_1$ in $\mathcal{E}$, if e_2 has a conflict with an edge e_2^*, add e_2^* to $\mathcal{F}''$. After this work has been performed on all the edges of $\mathcal{F}'$, collapse all of such edges, removing self-loops. We have the following:

  Lemma 10.8 *The edges of set $\mathcal{F}''$ do not have parallel edges.*

  Proof: Suppose, for a contradiction, that after Simplification 1 has been performed on all the edges of $\mathcal{F}'$, $\mathcal{F}''$ contains an edge $e_1 \in \mathcal{G}_i$ joining vertices u and v, such that there exists an edge $e_2 \in \mathcal{G}_i$ also joining vertices u and v. Since $e_1 \in \mathcal{F}''$, there exists an edge e_3 joining vertices w and x that has been removed when applying Simplification 1 to collapse an edge e_4 also joining vertices w and x. Consider the step s_i of Simplification 1 in which e_4 has been collapsed. Since e_3 cannot have a conflict with both e_1 and e_2, vertices w and x are before step s_i one inside and one outside the cycle composed of the edges e_1 and e_2 (see Fig. 10.13). Hence, before step s_i, e_4 either intersects e_1 or e_2, that gives us a contradiction, since e_4 is supposed to be a conflict-free edge, otherwise it would have not been collapsed during an application of Simplification 1. □

  Compute any spanning forest of the edges of $\mathcal{F}''$ and perform Simplification 1 on all the edges of such a forest. The above lemma guarantees that after this second pass no new conflict-free edge can be originated.

- *Separating edges.* After the end of the previous step, a set of current multigraphs $\mathcal{G}_i$'s is returned. Exploiting such multigraphs, a set $\mathcal{H}$ of

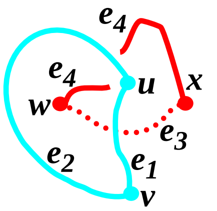


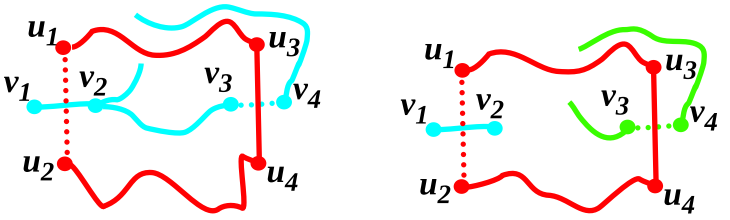
Figure 10.13: Illustration for the proof of Lemma 10.8

separating edges is constructed as follows. First, associate a name to each face of each multigraph $\mathcal{G}_i$. Second, for each edge e in each multigraph $\mathcal{G}_i$, record the names of the two faces incident to e. Third, for each edge e in each multigraph $\mathcal{G}_i$, verify if the faces incident to e are the same. If yes, then add e to $\mathcal{H}$. Observe that $\mathcal{H}$ is a set containing edges coming from all $\mathcal{G}_i$'s. Each edge e is labelled with a value indicating that e is a separating edge. This computation takes time linear in the number of edges in the $\mathcal{G}_i$'s.

After the set $\mathcal{H}$ has been created, for each edge e in $\mathcal{H}$, check if the edge e^* conflicting with e is a separating edge. If yes, return non-c-planar. Otherwise, delete e^* and collapse e. Observe that e has no parallel edges, otherwise it would not be a separating edge. After this has been done for all the edges in $\mathcal{H}$, it is easy to see that no conflict-free edge has been created. On the other hand, some edges in $\mathcal{G}_i$ could now be separating edges. However, if this happens, then we can conclude that C is not c-planar as stated in the following lemmas:

Lemma 10.9 *Consider a face f of $\mathcal{G}_i$. Suppose that f contains a separating pair composed by edges (u_1, u_2) and (u_3, u_4). Suppose that (u_1, u_2) has a conflict with an edge (v_1, v_2) that is a separating edge, and that (u_3, u_4) has a conflict with an edge (v_3, v_4). We have that C is not c-planar.*

Proof: Suppose w.l.o.g. that $(v_1, v_2) \in \mathcal{G}_j$ and that removing (u_1, u_2) and (u_3, u_4) disconnects $\mathcal{G}_i$ in two connected components $\mathcal{G}_i^1$ and $\mathcal{G}_i^2$ such that $u_1, u_3 \in \mathcal{G}_i^1$ and $u_2, u_4 \in \mathcal{G}_i^2$. By Lemma 10.5, C admits a saturator only if it admits an acyclic saturator S such that $(v_1, v_2) \in S$ and $(u_1, u_2) \notin S$. Since (u_1, u_2) and (u_3, u_4) compose a separating pair, $(u_3, u_4) \in S$,



(a) (b)

Figure 10.14: Illustrations for the proof of Lemma 10.9

otherwise no path in S could connect $\mathcal{G}_i^1$ and $\mathcal{G}_i^2$. Consider paths $S(u_1, u_3)$ and $S(u_2, u_4)$ connecting u_1 and u_3, and connecting u_2 and u_4 in S (such paths are single vertices if $u_1 = u_3$ and/or $u_2 = u_4$).

If $(v_3, v_4) \in \mathcal{G}_j$ (see Fig. 10.14 (a)), then let v_1 (v_2) be the one out of v_1 and v_2 that is outside (resp. inside) f and let v_4 (v_3) be the one out of v_3 and v_4 that is outside (resp. inside) f. Then, v_3 (v_4) is inside (resp. outside) cycle $\mathcal{C} = S(u_1, u_3) \cup (u_3, u_4) \cup S(u_2, u_4) \cup (u_1, u_2)$. Consider path $S(v_2, v_3)$ connecting v_2 and v_3 in S. Notice that $S(v_2, v_3)$ lies completely inside $\mathcal{C}$, otherwise $S(v_2, v_3)$ would contain an edge conflicting with an edge of $S(u_1, u_3)$, or an edge conflicting with an edge of $S(u_2, u_4)$, or an edge conflicting with (u_3, u_4), implying that S contains two conflicting edges. Consider path $S(v_2, v_4)$ connecting v_2 and v_4 in S. Since v_2 is inside $\mathcal{C}$ and v_4 is outside $\mathcal{C}$, $S(v_2, v_4)$ lies in part inside and in part outside $\mathcal{C}$. It follows that either there exists an edge of $S(v_2, v_4)$ conflicting with an edge of $S(u_1, u_3)$, or an edge conflicting with an edge of $S(u_2, u_4)$, or an edge conflicting with (u_3, u_4), implying that S contains two conflicting edges, or $S(v_2, v_4)$ contains edge (v_1, v_2). However, since (v_1, v_2) is a separating edge no path excluding (v_1, v_2) and connecting v_1 to v_4 exists in $\mathcal{G}_i$.

If $(v_3, v_4) \in \mathcal{G}_k$, with $k \neq i, j$, then vertices v_3 and v_4 lie one inside and one outside $\mathcal{C}$. Hence, any path connecting v_3 and v_4 in S either contains an edge conflicting with an edge of $S(u_1, u_3)$, or with an edge of $S(u_2, u_4)$, or with (u_3, u_4) implying that S contains two conflicting edges, or contains an edge conflicting with (u_1, u_2), implying that (u_1, u_2) has two conflicting edges, respectively. □

Lemma 10.10 *Suppose that each edge of $\mathcal{H}$ has a conflict with a non-separating edge. Collapse the edges in $\mathcal{H}$, repeatedly applying Simplification 2. Either the resulting multigraphs $\mathcal{G}_i$ are edge 2-connected or C is not c-planar.*

Proof: Order the edges of $\mathcal{H}$ in whichever way $\{e_1, e_2, \dots, e_k\}$. Let e_j, with $1 \leq j \leq k$, be the first edge in $\{e_1, e_2, \dots, e_k\}$ such that (i) collapsing edges $e_1, e_2, \dots, e_{j-1}$ from the $\mathcal{G}_i$'s does not create new separating edges and (ii) collapsing edge e_j creates a new separating edge $\overline{e}_j$. Suppose, for a contradiction, that a saturator of C exists. Then, by Lemma 10.5, there exists a saturator S containing edges $e_1, e_2, \dots, e_{j-1}$ and not containing the edges that have conflicts with edges $e_1, e_2, \dots, e_{j-1}$. Consider the $\mathcal{G}_i$'s after edges $e_1, e_2, \dots, e_{j-1}$ have been collapsed (and the edges that have conflicts with edges $e_1, e_2, \dots, e_{j-1}$ have been removed). Refer to Fig. 10.14. Since collapsing edge $e_j = (v_1, v_2)$ creates a new separating edge (u_3, u_4), (u_1, u_2) and (u_3, u_4) compose a separation pair for a multigraph $\mathcal{G}_i$, where (u_1, u_2) is the edge that has a conflict with (v_1, v_2). Hence, there exists a face of $\mathcal{G}_i$ containing (u_1, u_2) and (u_3, u_4). Since no edge (and hence neither (u_3, u_4)) is conflict-free, the statement follows from Lemma 10.9. □

After the collapse of all the edges in $\mathcal{H}$ and the removal of their conflicting edges, a set of current multigraphs $\mathcal{G}_i$'s is returned. Exploiting such multigraphs, Condition 3 can be tested as follows. First, associate a name to each face of each multigraph $\mathcal{G}_i$; second, for each edge e in each multigraph $\mathcal{G}_i$, record the names of the two faces incident to e, and third, for each edge e in each multigraph $\mathcal{G}_i$, verify if the faces incident to e are the same. If this is true for at least one edge, by the previous lemmas we can return that the input graph is not c-planar, otherwise the current $\mathcal{G}_i$'s are the candidate saturating graphs of the clusters.

For each pair of distinct clusters μ_i and μ_j, we check if $\mathcal{G}^*_{i,j}$ is edge 2-connected (Condition 4 of Theorem 10.3) and if $\mathcal{G}^*_{i,j}$ is dual to $\mathcal{G}^*_{j,i}$ (Condition 5 of Theorem 10.3). This is easily done in linear time because of the following property.

Property 10.7 $\sum_{i,j} |\mathcal{G}^*_{i,j}| = O(n)$, *where* $|\mathcal{G}^*_{i,j}|$ *is the size of the graph.*

Proof: It trivially follows from Property 10.6. □

Hence, we can conclude the section with the following theorem.

Theorem 10.4 *The c-planarity of an n-vertex embedded flat clustered graph $C(G,T)$ with at most five vertices per face can be tested in $O(n)$ time and space.*

As a consequence of the arguments discussed in this section, we remark that a theorem analogous to Theorem 10.4 holds even for single-conflict clustered graphs.

10.5 Conclusions and Open Problems

In this chapter we have considered the problem of testing the c-planarity of a clustered graph. We have shown that the c-planarity of embedded flat clustered graphs with at most five vertices per face and, more generally, the c-planarity of single-conflict embedded flat clustered graphs can be efficiently tested.

We remark that the simplification phase described in Sect. 10.3 is a preprocessing that can be performed on any embedded flat clustered graph. This allows to reduce the problem of testing the c-planarity of such graphs to the one of deciding whether a set of edge 2-connected candidate saturating graphs admits a set of non-conflicting spanning trees. However, it's rather easy to see that the characterization shown in Theorem 10.2 does not hold for general embedded flat clustered graphs.

We conclude by providing a list of families of embedded clustered graphs for which, in our opinion, determining the time complexity of a c-planarity testing is worth of interest: (i) single-conflict general (non-flat) embedded clustered graphs; (ii) embedded flat clustered graphs such that for each face of the underlying graph there are at most two (or a constant number of) vertices of the same cluster; and (iii) embedded flat clustered graphs.

Part V

Publications and Bibliography

Other Research Activities

Simultaneously with the research for the development of this thesis, other topics in the area of Graph Drawing have been dealt with:

- *Simultaneous Graph Drawing.* A *simultaneous embedding* of a set of n-vertex graphs $G_1, G_2, \ldots, G_k$ is a set of planar drawings $\Gamma_1, \Gamma_2, \ldots, \Gamma_k$ of $G_1, G_2, \ldots, G_k$ on the same set of n points. Different constraints have been considered, namely the ones in which the mapping between the vertices of the graph is given or not (respectively, *simultaneous embedding with mapping* and *simultaneous embedding without mapping*), the ones in which the edges of the graph are forced to be drawn as straight-lines or not (respectively, *geometric simultaneous embedding* and *simultaneous embedding*), the ones in which edges shared by different graphs are forced to have the same drawing or not (respectively, *simultaneous embedding with fixed edges* and *simultaneous embedding*). The typical problems that are studied in this field are the combinatorial determination of the classes of graphs that always have a simultaneous embedding, and the complexity of determining whether two or more graphs have a simultaneous embedding.

- *Planar Packing.* A *packing* of a set of n-vertex graphs $G_1, G_2, \ldots, G_k$ is a graph G containing graphs $G_1, G_2, \ldots, G_k$ as edge-disjoint subgraphs. A *planar packing* is such that G is planar. The work in the area of planar packing mainly concentrates around the conjecture that every two non-star trees admit a planar packing.

- *Straight-line Embeddings into Point Sets.* A *straight-line embedding* of a graph G into a point set P is a mapping of the vertices of G to the points of P such that the resulting straight-line drawing of G is planar. Typical questions in this area ask for the largest class of graphs that admit a straight-line embedding into every point set, and for the complexity of testing whether a graph has a straight-line embedding into a given point set. Recently, the same question referred to directed graphs and *upward straight-line embeddings* has been studied.

- *Bad Graph Drawing.* A *bad graph drawing* is a drawing as unreadable as possible. Such drawings are clearly useless for applications, however they turn out to have nice combinatorial properties. Drawings with the maximum number of bends and drawings in which all the faces are represented by non-convex polygons are examples of bad drawings.

Publications

Journal Publications

1. G. Di Battista, F. Frati. Efficient C-Planarity Testing for Embedded Flat Clustered Graphs with Small Faces. *Journal of Graph Algorithms and Applications*, *Special Issue of Selected Papers from GD '07*, 2009. To appear.

2. F. Frati, M. Kaufmann, S. Kobourov. Constrained Simultaneous and Near Simultaneous Embeddings. *Journal of Graph Algorithms and Applications*, *Special Issue of Selected Papers from GD '07*, 2009. To appear.

3. P. F. Cortese, G. Di Battista, F. Frati, M. Patrignani, M. Pizzonia. C-Planarity of C-Connected Clustered Graphs. *Journal of Graph Algorithms and Applications*, 12(2):225-262. November 2008.

4. G. Di Battista, G. Drovandi, F. Frati. How to Draw a Clustered Tree. *Journal of Discrete Algorithms*, 2009. To appear.

5. F. Frati, M. Geyer, M. Kaufmann. Planar Packings of Trees and Spider Trees. *Information Processing Letters*, 109(6):301-307. February 2009.

6. G. Di Battista, F. Frati, M. Patrignani. On Embedding a Graph on the Grid with the Maximum Number of Bends and Other Bad Features. *Theory of Computing Systems*, *Special Issue of Selected Papers from FUN '07*, 44(2):143-149. February 2009.

7. F. Frati. On Minimum Area Planar Upward Drawings of Directed Trees and Other Families of Directed Acyclic Graphs. *International Journal of Computational Geometry and Applications*, 18(3):251-271. June 2008.

8. G. Di Battista, F. Frati. Small Area Drawings of Outerplanar Graphs. *Algorithmica*, 2009. To appear.

Conference Publications

1. P. Angelini, F. Frati, L. Grilli. An Algorithm to Construct Greedy Drawings of Triangulations. *In 16th International Symposium on Graph Drawing (GD '08),* volume 5417 of Lecture Notes Comput. Sci., pages 26-37, 2008.

2. G. Di Battista, F. Frati, M. Patrignani. Non-Convex Representations of Graphs. *In 16th International Symposium on Graph Drawing (GD '08),* volume 5417 of Lecture Notes Comput. Sci., pages 390-395, 2008.

3. F. Frati. A Lower Bound on the Area Requirements of Series-Parallel Graphs. *In 34th International Workshop on Graph-Theoretic Concepts in Computer Science (WG '08),* volume 5344 of Lecture Notes Comput. Sci., pages 159-170, 2008.

4. Alejandro Estrella-Balderrama, F. Frati, S. Kobourov. Upward Straight-line Embeddings of Directed Graphs into Point Sets. *In 34th International Workshop on Graph-Theoretic Concepts in Computer Science (WG '08),* volume 5344 of Lecture Notes Comput. Sci., pages 122-133, 2008.

5. F. Frati. Straight-line Orthogonal Drawings of Binary and Ternary Trees. *In 15th International Symposium on Graph Drawing (GD '07),* volume 4875 of Lecture Notes Comput. Sci., pages 76-87, 2007.

6. F. Frati, M. Kaufmann, S. Kobourov. Constrained Simultaneous and Near Simultaneous Embeddings. *In 15th International Symposium on Graph Drawing (GD '07),* volume 4875 of Lecture Notes Comput. Sci., pages 268-279, 2007.

7. G. Di Battista, F. Frati. Efficient C-Planarity Testing for Embedded Flat Clustered Graphs with Small Faces. *In 15th International Symposium on Graph Drawing (GD '07),* volume 4875 of Lecture Notes Comput. Sci., pages 291-302, 2007.

8. F. Frati, M. Patrignani. A Note on Minimum Area Straight-line Drawings of Planar Graphs. *In 15th International Symposium on Graph Drawing*

(GD '07), volume 4875 of Lecture Notes Comput. Sci., pages 339-344, 2007.

9. F. Frati. Straight-line Drawings of Outerplanar Graphs in O(dn log n) Area. *In 19th Canadian Conference on Computational Geometry (CCCG '07),* pages 225-228, 2007.

10. F. Frati, M. Geyer, M. Kaufmann. Packing and Squeezing Subgraphs into Planar Graphs. *In 32nd International Symposium on Mathematical Foundations of Computer Science (MFCS '07),* volume 4708 of Lecture Notes Comput. Sci., pages 394-405, 2007.

11. F. Frati. On Minimum Area Planar Upward Drawings of Directed Trees and Other Families of Directed Acyclic Graphs. *In 33rd International Workshop on Graph-Theoretic Concepts in Computer Science (WG '07),* volume 4769 of Lecture Notes Comput. Sci., pages 133-144, 2007.

12. G. Di Battista, G. Drovandi, F. Frati. How to Draw a Clustered Tree. *In 10th Workshop on Algorithms and Data Structures (WADS '07),* volume 4619 of Lecture Notes Comput. Sci., pages 89-101, 2007.

13. U. Brandes, C. Erten, J. Fowler, F. Frati, M. Geyer, C. Gutwenger, S.-H. Hong, M. Kaufmann, S. Kobourov, G. Liotta, P. Mutzel, A. Symvonis. Colored Simultaneous Geometric Embeddings. *In 13th Annual International Computing and Combinatorics Conference (COCOON '07),* volume 4598 of Lecture Notes Comput. Sci., pages 254-263, 2007.

14. G. Di Battista, F. Frati, M. Patrignani. On Embedding a Graph on the Grid with the Maximum Number of Bends and Other Bad Features. *In Fun with Algorithms (FUN '07),* volume 4475 of Lecture Notes Comput. Sci., pages 1-13, 2007. (bibtex)

15. P. F. Cortese, G. Di Battista, F. Frati, L. Grilli, K. A. Lehmann, G. Liotta, M. Patrignani, I. Tollis, F. Trotta. On the Topologies of Local Minimum Spanning Trees. *In 3rd Workshop on Combinatorial and Algorithmic Aspects of the Networks (CAAN '06),* volume 4235 of Lecture Notes Comput. Sci., pages 31-44, 2006.

16. F. Frati, G. Di Battista. Three Dimensional Drawings of Bounded Degree Trees. *In 14th International Symposium on Graph Drawing (GD '06),* volume 4372 of Lecture Notes Comput. Sci., pages 89-94, 2006.

17. F. Frati. Embedding Graphs Simultaneously with Fixed Edges. *In 14th International Symposium on Graph Drawing (GD '06),* volume 4372 of Lecture Notes Comput. Sci., pages 108-113, 2006.

18. G. Di Battista, F. Frati. Small Area Drawings of Outerplanar Graphs. *In 13th International Symposium on Graph Drawing (GD '05),* volume 3843 of Lecture Notes Comput. Sci., pages 89-100, 2005.

Technical Reports

1. G. Di Battista, F. Frati, M. Patrignani. Non-Convex Representations of Graphs. Technical Report RT-DIA-134-2008, Dept. of Computer Science and Automation, Roma Tre University, 2008.
   http://dipartimento.dia.uniroma3.it/ricerca/rapporti/rt/2008-134.pdf

2. A. Estrella-Balderrama, F. Frati, S. Kobourov. Upward Straight-line Embeddings of Directed Graphs into Point Sets. Technical Report RT-DIA-133-2008, Dept. of Computer Science and Automation, Roma Tre University, 2008.
   http://dipartimento.dia.uniroma3.it/ricerca/rapporti/rt/2008-133.pdf

3. F. Frati, M. Kaufmann. Polynomial Area Bounds for MST Embeddings of Trees. Technical Report RT-DIA-122-2008, Dept. of Computer Science and Automation, Roma Tre University, 2008.
   http://dipartimento.dia.uniroma3.it/ricerca/rapporti/rt/2008-122.pdf

4. F. Frati, M. Kaufmann, S. Kobourov. *Constrained Simultaneous and Near-Simultaneous Embeddings.* Technical Report RT-DIA-120-2007, Dept. of Computer Science and Automation, University of Roma Tre, 2007.
   http://dipartimento.dia.uniroma3.it/ricerca/rapporti/rt/2007-120.pdf

5. G. Di Battista, F. Frati. *Efficient C-Planarity Testing for Embedded Flat Clustered Graphs with Small Faces.* Technical Report RT-DIA-119-2007, Dept. of Computer Science and Automation, University of Roma Tre, 2007.
   http://dipartimento.dia.uniroma3.it/ricerca/rapporti/rt/2007-119.pdf

6. G. Di Battista, G. Drovandi, F. Frati. *How to Draw a Clustered Tree.* Technical Report RT-DIA-115-2007, Dept. of Computer Science and Automation, University of Roma Tre, 2007.
   http://dipartimento.dia.uniroma3.it/ricerca/rapporti/rt/2007-115.pdf

7. F. Frati, M. Geyer, M. Kaufmann. *Packing and Squeezing Subgraphs into Planar Graphs.* Technical Report RT-DIA-114-2007, Dept. of Computer Science and Automation, University of Roma Tre, 2007.
http://dipartimento.dia.uniroma3.it/ricerca/rapporti/rt/2007-114.pdf

8. P. F. Cortese, G. Di Battista, F. Frati, L. Grilli, K. A. Lehmann, G. Liotta, M. Patrignani, I. Tollis, F. Trotta. *On the Topologies of Local Minimum Spanning Trees.* Technical Report RT-001-06, Dip. Ingegneria Elettronica e dell'Informazione, Univ. Perugia, 2006.
http://www.diei.unipg.it/rt/RT-001-06-Cortese-DiBattista-Frati-Grilli-Lehmann-Liotta-Patrignani-Tollis-Trotta.pdf

9. P. F. Cortese, G. Di Battista, F. Frati, M. Patrignani, M. Pizzonia. *C-Planarity of C-Connected Clustered Graphs: Part I - Characterization.* Technical Report RT-DIA-109-2006, Dip. Informatica e Automazione, Univ. Roma Tre, 2006
http://dipartimento.dia.uniroma3.it/ricerca/rapporti/rt/2006-109.pdf

10. P. F. Cortese, G. Di Battista, F. Frati, M. Patrignani, M. Pizzonia. *C-Planarity of C-Connected Clustered Graphs: Part II - Testing and Embedding Algorithm.* Technical Report RT-DIA-110-2006, Dip. Informatica e Automazione, Univ. Roma Tre, 2006
http://dipartimento.dia.uniroma3.it/ricerca/rapporti/rt/2006-110.pdf

Bibliography

[AC04] S. Arora and K. L. Chang. Approximation schemes for degree-restricted MST and red-blue separation problems. *Algorithmica*, 40(3):189–210, 2004.

[AFG08] P. Angelini, F. Frati, and L. Grilli. An algorithm to construct greedy drawings of triangulations. In M. Patrignani and I. Tollis, editors, *Graph Drawing (GD '08)*, LNCS, pages 26–37, 2008.

[AHU83] A. V. Aho, J. E. Hopcroft, and J. D. Ullman. *Data Structures and Algorithms*. Addison-Wesley, Reading, MA, 1983.

[And63] G. E. Andrews. A lower bound for the volumes of stricly convex bodies with many boundary points. *Trans. AMS*, 106:270–279, 1963.

[Aro98] S. Arora. Polynomial time approximation schemes for euclidean traveling salesman and other geometric problems. *J. ACM*, 45(5):753–782, 1998.

[BB05] T. C. Biedl and F. J. Brandenburg. Drawing planar bipartite graphs with small area. In *Canadian Conference on Computational Geometry (CCCG '05)*, pages 105–108, 2005.

[BCB$^+$94] P. Bertolazzi, R. F. Cohen, G. Di Battista, R. Tamassia, and I. G. Tollis. How to draw a series-parallel digraph. *Int. J. Comp. Geom. Appl.*, 4(4):385–402, 1994.

[BCGG06] M. Ben-Chen, C. Gotsman, and S. J. Gortler. Routing with guaranteed delivery on virtual coordinates. In *Canadian Conference on Computational Geometry (CCCG '06)*, 2006.

[BCGW07] M. Ben-Chen, C. Gotsman, and C. Wormser. Distributed computation of virtual coordinates. In J. Erickson, editor, *Symposium on Computational Geometry (SoCG ’07)*, pages 210–219, 2007.

[BCLO03] T. C. Biedl, T. M. Chan, and A. López-Ortiz. Drawing $K_{2,n}$: A lower bound. *Inf. Process. Lett.*, 85(6):303–305, 2003.

[BDLM94] P. Bertolazzi, G. Di Battista, G. Liotta, and C. Mannino. Upward drawings of triconnected digraphs. *Algorithmica*, 12(6):476–497, 1994.

[BF07] G. Di Battista and F. Frati. Efficient c-planarity testing for embedded flat clustered graphs with small faces. In S. H. Hong, T. Nishizeki, and W. Quan, editors, *Graph Drawing (GD ’07)*, volume 4875 of *LNCS*, pages 291–302, 2007.

[BFM07] N. Bonichon, S. Felsner, and M. Mosbah. Convex drawings of 3-connected plane graphs. *Algorithmica*, 47(4):399–420, 2007.

[Bie02] T. C. Biedl. Drawing outer-planar graphs in $O(n \log n)$ area. In M. T. Goodrich, editor, *Graph Drawing (GD ’02)*, volume 2528 of *LNCS*, pages 54–65, 2002.

[Bie05] T. C. Biedl. Small poly-line drawings of series-parallel graphs. Tech. Report CS-2007-23, School of Computer Science, University of Waterloo, Canada, 2005.

[BKK97] T. C. Biedl, G. Kant, and M. Kaufmann. On triangulating planar graphs under the four-connectivity constraint. *Algorithmica*, 19(4):427–446, 1997.

[BL76] K. Booth and G. Lueker. Testing for the consecutive ones property interval graphs and graph planarity using PQ-tree algorithms. *J. Comp. Syst. Sci.*, 13:335–379, 1976.

[BLL96] P. Bose, W. Lenhart, and G. Liotta. Characterizing proximity trees. *Algorithmica*, 16(1):83–110, 1996.

[BM76] J. A. Bondy and U. S. R. Murty. *Graph Theory with Applications*. Macmillan, London, United Kingdom, 1976.

[Bos02] P. Bose. On embedding an outer-planar graph in a point set. *Computat. Geom. Th. Appl.*, 23(3):303–312, 2002.

[BP92] I. Bárány and J. Pach. On the number of convex lattice polygons. *Combinatorics, Probability & Computing*, 1:295–302, 1992.

[BR06] I. Bárány and G. Rote. Strictly convex drawings of planar graphs. *Documenta Math.*, 11:369–391, 2006.

[Bra08] F. J. Brandenburg. Drawing planar graphs on $\frac{8}{9}n^2$ area. *Elet. Notes Discr. Math.*, 31:37–40, 2008.

[BSM02] N. Bonichon, B. Le Saëc, and M. Mosbah. Optimal area algorithm for planar polyline drawings. In L. Kucera, editor, *Graph-Theoretic Concepts in Computer Science (WG '02)*, volume 2573 of *LNCS*, pages 35–46, 2002.

[BT04] I. Bárány and N. Tokushige. The minimum area of convex lattice n-gons. *Combinatorica*, 24(2):171–185, 2004.

[CB05] P. F. Cortese and G. Di Battista. Clustered planarity. In *Symposium on Computational Geometry (SoCG '05)*, pages 32–34, 2005.

[CBF$^+$08] P. F. Cortese, G. Di Battista, F. Frati, M. Patrignani, and M. Pizzonia. C-planarity of c-connected clustered graphs. *J. Graph Alg. Appl.*, 12(2):225–262, 2008.

[CBP92] P. Crescenzi, G. Di Battista, and A. Piperno. A note on optimal area algorithms for upward drawings of binary trees. *Computat. Geom. Th. Appl.*, 2:187–200, 1992.

[CDGK01] C. C. Cheng, C. A. Duncan, M. T. Goodrich, and S. G. Kobourov. Drawing planar graphs with circular arcs. *Discr. Computat. Geom.*, 25(3):405–418, 2001.

[CDPP05a] P. F. Cortese, G. Di Battista, M. Patrignani, and M. Pizzonia. Clustering cycles into cycles of clusters. *J. Graph Alg. Appl.*, 9(3):391–413, 2005.

[CDPP05b] P. F. Cortese, G. Di Battista, M. Patrignani, and M. Pizzonia. On embedding a cycle in a plane graph. In P. Healy and N. S. Nikolov, editors, *Graph Drawing (GD '05)*, volume 3843 of *LNCS*, pages 49–60, 2005.

[CGKT02] T. M. Chan, M. T. Goodrich, S. Rao Kosaraju, and R. Tamassia. Optimizing area and aspect ratio in straight-line orthogonal tree drawings. *Computat. Geom. Th. Appl.*, 23(2):153–162, 2002.

[CGT96] M. Chrobak, M. T. Goodrich, and R. Tamassia. Convex drawings of graphs in two and three dimensions (preliminary version). In *Symposium on Computational Geometry (SoCG '96)*, pages 319–328, 1996.

[Cha02] T.M. Chan. A near-linear area bound for drawing binary trees. *Algorithmica*, 34(1):1–13, 2002.

[Cha04] T. M. Chan. Euclidean bounded-degree spanning tree ratios. *Discr. Computat. Geom.*, 32(2):177–194, 2004.

[CK97] M. Chrobak and G. Kant. Convex grid drawings of 3-connected planar graphs. *Int. J. Comp. Geom. Appl.*, 7(3):211–223, 1997.

[CLJ06] H. Cheng, Q. Liu, and X. Jia. Heuristic algorithms for real-time data aggregation in wireless sensor networks. In S. Onoe, M. Guizani, H.-H. Chen, and M. Sawahashi, editors, *Wireless Communications and Mobile Computing Conference (IWCMC '08)*, pages 1123–1128, 2006.

[CLRS01] T. H. Cormen, C. E. Leiserson, R. L. Rivest, and C. Stein. *Introduction to Algorithms*. McGraw-Hill Book Company, Boston, MA, 2001.

[CN98] M. Chrobak and S.-I. Nakano. Minimum-width grid drawings of plane graphs. *Computat. Geom. Th. Appl.*, 11:29–54, 1998.

[CP95] M. Chrobak and T. H. Payne. A linear-time algorithm for drawing a planar graph on a grid. *Inf. Process. Lett.*, 54(4):241–246, 1995.

[CW06] S. Cornelsen and D. Wagner. Completely connected clustered graphs. *J. Discrete Algorithms*, 4(2):313–323, 2006.

[CYN84] N. Chiba, T. Yamanouchi, and T. 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] E. Dahlhaus. A linear time algorithm to recognize clustered graphs and its parallelization. In C. L. Lucchesi and A. V. Moura, editors, *Latin American Theoretical Informatics (LATIN '98)*, LNCS, pages 239–248, 1998.

[DDF07] G. Di Battista, G. Drovandi, and F. Frati. How to draw a clustered tree. In F. K. H. A. Dehne, J. R. Sack, and N. Zeh, editors, *Algorithms and Data Structures (WADS '07)*, volume 4619 of *LNCS*, pages 89–101, 2007.

[DDF09] G. Di Battista, G. Drovandi, and F. Frati. How to draw a clustered tree. *J. Discrete Algorithms*, 2009. To appear.

[Del34] B. Delaunay. Sur la sphére vide. *Bull. Acad. Sci. USSR (VII), Classe Sci. Mat. Nat.*, 7:793–800, 1934.

[DETT99] G. Di Battista, P. Eades, R. Tamassia, and I. G. Tollis. *Graph Drawing*. Prentice Hall, Upper Saddle River, NJ, 1999.

[DF05] G. Di Battista and F. Frati. Small area drawings of outerplanar graphs. In Healy and Nikolov, editors, *Graph Drawing (GD '05)*, volume 3843 of *LNCS*, pages 89–100, 2005.

[DF09] G. Di Battista and F. Frati. Small area drawings of outerplanar graphs. *Algorithmica*, 2009. To appear.

[Dha08] R. Dhandapani. Greedy drawings of triangulations. In S. T. Huang, editor, *Symposium on Discrete Algorithms (SODA '08)*, pages 102–111, 2008.

[Die05] R. Diestel. *Graph Theory*. Springer, Heidelberg, Germany, 2005.

[Dil90] M. B. Dillencourt. Realizability of Delaunay triangulations. *Inf. Process. Lett.*, 33(6):283–287, 1990.

[DLL94] G. Di Battista, W. Lenhart, and G. Liotta. Proximity drawability: a survey. In R. Tamassia and I. G. Tollis, editors, *Graph Drawing (GD '94)*, volume 894 of *LNCS*, pages 328–339, 1994.

[DLMW05] E. Di Giacomo, G. Liotta, H. Meijer, and S. K. Wismath. Volume requirements of 3D upward drawings. In P. Healy and N. S. Nikolov, editors, *Graph Drawing (GD '05)*, pages 101–110, 2005.

[DLR90] G. Di Battista, W.P. Liu, and I. Rival. Bipartite graphs, upward drawings, and planarity. *Inf. Process. Lett.*, 36(6):317–322, 1990.

[DLW06] G. Di Battista, G. Liotta, and S. Whitesides. The strength of weak proximity. *J. Discrete Algorithms*, 4(3):384–400, 2006.

[dPP88] H. de Fraysseix, J. Pach, and R. Pollack. Small sets supporting fáry embeddings of planar graphs. In *Symposium on Theory of Computing (STOC '88)*, pages 426–433, 1988.

[dPP90] H. de Fraysseix, J. Pach, and R. Pollack. How to draw a planar graph on a grid. *Combinatorica*, 10(1):41–51, 1990.

[dR82] H. de Fraysseix and P. Rosenstiehl. A depth-first-search characterization of planarity. *Ann. Discr. Math.*, 13:75–80, 1982.

[DS96] M. B. Dillencourt and W. D. Smith. Graph-theoretical conditions for inscribability and Delaunay realizability. *Discr. Math.*, 161(1-3):63–77, 1996.

[DT81] D. Dolev and H. W. Trickey. On linear area embedding of planar graphs. Technical report, Stanford University, Stanford, USA, 1981.

[DT88] G. Di Battista and R. Tamassia. Algorithms for plane representations of acyclic digraphs. *Theor. Comput. Sci.*, 61:175–198, 1988.

[DT90] G. Di Battista and R. Tamassia. On-line graph algorithms with SPQR-trees. In M. Paterson, editor, *International Colloquium on Automata, Languages and Programming (ICALP '90)*, LNCS, pages 598–611, 1990.

[DT96a] G. Di Battista and R. Tamassia. On-line maintenance of triconnected components with spqr-trees. *Algorithmica*, 15(4):302–318, 1996.

[DT96b] G. Di Battista and R. Tamassia. On-line planarity testing. *SIAM J. Comput.*, 25(5):956–997, 1996.

[DTL99] G. Di Battista, R. Tamassia, and Vismara L. Output-sensitive reporting of disjoint paths. *Algorithmica*, 23(4):302–340, 1999.

[DTT92] G. Di Battista, R. Tamassia, and I. G. Tollis. Area requirement and symmetry display of planar upward drawings. *Disc. Computat. Geom.*, 7:381–401, 1992.

[DTV01] G. Di Battista, R. Tamassia, and L. Vismara. Incremental convex planarity testing. *Inf. Comput.*, 169(1):94–126, 2001.

[dvKOS00] M. de Berg, M. van Kreveld, M. Overmars, and O. Schwarzkopf. *Computational Geometry - Algorithms and Applications*. Springer, Heidelberg, Germany, 2000.

[Ede87] H. Edelsbrunner. *Algorithms in Combinatorial Geometry*. Springer, New York, NY, 1987.

[EFLN06] P. Eades, Q. Feng, X. Lin, and H. Nagamochi. Straight-line drawing algorithms for hierarchical graphs and clustered graphs. *Algorithmica*, 44(1):1–32, 2006.

[EFN99] P. Eades, Q. Feng, and H. Nagamochi. Drawing clustered graphs on an orthogonal grid. *J. Graph Alg. Appl.*, 3(4):3–29, 1999.

[EG08] D. Eppstein and M. Goodrich. Succinct greedy graph drawing in the hyperbolic plane. In M. Patrignani and I. Tollis, editors, *Graph Drawing 2008*, pages 14–25, 2008.

[Epp92] D. Eppstein. Parallel recognition of series-parallel graphs. *Inf. Comput.*, 98(1):41–55, 1992.

[ET76] S. Even and R. E. Tarjan. Computing an st-numbering. *Theor. Comp. Sci.*, 2:339–344, 1976.

[Eve79] S. Even. *Graph Algorithms*. Computer Science Press, Potomac, Maryland, 1979.

[EW96a] P. Eades and S. Whitesides. The logic engine and the realization problem for nearest neighbor graphs. *Theor. Comput. Sci.*, 169(1):23–37, 1996.

[EW96b] P. Eades and S. Whitesides. The realization problem for euclidean minimum spanning trees in NP-hard. *Algorithmica*, 16(1):60–82, 1996.

[Far48] I. Fary. On straight line representions of planar graphs. *Acta. Sci. Math.*, 11:229–233, 1948.

[FCE95a] Q. Feng, R. F. Cohen, and P. Eades. How to draw a planar clustered graph. In D. Du and M. Li, editors, *Computing and Combinatorics Conference (COCOON '95)*, pages 21–30, 1995.

[FCE95b] Q. Feng, R. F. Cohen, and P. Eades. Planarity for clustered graphs. In P. G. Spirakis, editor, *European Symposium on Algorithms (ESA '95)*, volume 979 of *LNCS*, pages 213–226, 1995.

[Fen97] Q. Feng. *Algorithms for Drawing Clustered Graphs*. PhD thesis, The University of Newcastle, Australia, 1997.

[FK08] F. Frati and M. Kaufmann. Polynomial area bounds for MST embeddings of trees. Tech. Report RT-DIA-122-2008, Dept. of Computer Science and Automation, Roma Tre University, 2008.

[FLW03] S. Felsner, G. Liotta, and S. K. Wismath. Straight-line drawings on restricted integer grids in two and three dimensions. *J. Graph Alg. Appl.*, 7(4):363–398, 2003.

[FP07] F. Frati and M. Patrignani. A note on minimum area straight-line drawings of planar graphs. In S. H. Hong and T. Nishizeki, editors, *Graph Drawing (GD '07)*, LNCS, pages 339–344, 2007.

[Fra07a] F. Frati. On minimum area planar upward drawings of directed trees and other families of directed acyclic graphs. In A. Brandstädt, D. Kratsch, and H. Müller, editors, *Graph-Theoretic Concepts in Computer Science (WG '07)*, volume 4769 of *LNCS*, pages 133–144, 2007.

[Fra07b] F. Frati. Straight-line drawings of outerplanar graphs in o(dn log n) area. In P. Bose, editor, *Canadian Conference on Computational Geometry (CCCG '07)*, pages 225–228, 2007.

[Fra07c] F. Frati. Straight-line orthogonal drawings of binary and ternary trees. In S. H. Hong, T. Nishizeki, and W. Quan, editors, *Graph Drawing (GD '07)*, volume 4875 of *LNCS*, pages 76–87, 2007.

[Fra08a] F. Frati. A lower bound on the area requirements of series-parallel graphs. In H. Broersma and T. Erlebach, editors, *Graph-Theoretic*

Concepts in Computer Science (WG '08), volume 5344 of *LNCS*, pages 159–170, 2008.

[Fra08b] F. Frati. On minimum area planar upward drawings of directed trees and other families of directed acyclic graphs. *Int. J. Comput. Geom. Appl.*, 18(3):251–271, 2008.

[GDLW06] E. Di Giacomo, W. Didimo, G. Liotta, and S. K. Wismath. Book embeddability of series-parallel digraphs. *Algorithmica*, 45(4):531–547, 2006.

[GGT96] A. Garg, M. T. Goodrich, and R. Tamassia. Planar upward tree drawings with optimal area. *Int. J. Comput. Geom. Appl.*, 6(3):333–356, 1996.

[Gia03] E. Di Giacomo. Drawing series-parallel graphs on restricted integer 3d grids. In G. Liotta, editor, *Graph Drawing (GD '03)*, volume 2912 of *LNCS*, pages 238–246, 2003.

[GJ79] M. R. Garey and D. S. Johnson. *Computers and Intractability: A Guide to the Theory of NP-Completeness.* W. H. Freeman, 1979.

[GJL$^+$02] C. Gutwenger, M. Jünger, S. Leipert, P. Mutzel, M. Percan, and R. Weiskircher. Advances in c-planarity testing of clustered graphs. In S. G. Kobourov and M. T. Goodrich, editors, *Graph Drawing (GD '02)*, volume 2528 of *LNCS*, pages 220–235, 2002.

[GLS05] M. T. Goodrich, G. S. Lueker, and J. Z. Sun. C-planarity of extrovert clustered graphs. In P. Healy and N. Nikolov, editors, *Graph Drawing (GD '05)*, volume 3843 of *LNCS*, pages 211–222, 2005.

[GM98] C. Gutwenger and P. Mutzel. Planar polyline drawings with good angular resolution. In S. Whitesides, editor, *Graph Drawing (GD '98)*, volume 1547 of *LNCS*, pages 167–182, 1998.

[Gon05] D. Gonçalves. Edge partition of planar graphs into two outerplanar graphs. In H. N. Gabow and R. Fagin, editors, *Symposium on Theory of Computing (STOC '05)*, pages 504–512. ACM, 2005.

[GPP91] P. Gritzmann, B. Mohar J. Pach, and R. Pollack. Embedding a planar triangulation with vertices at specified positions. *Amer. Math. Monthly*, 98:165–166, 1991.

[GR94] Z. Gao and R. B. Richter. 2-walks in circuit graphs. *J. Comb. Theory, Ser. B*, 62(2):259–267, 1994.

[GR02] Ashim Garg and Adrian Rusu. Straight-line drawings of binary trees with linear area and arbitrary aspect ratio. In S. G. Kobourov and M. T. Goodrich, editors, *Graph Drawing (GD '02)*, volume 2528 of *LNCS*, pages 320–331, 2002.

[GR03a] A. Garg and A. Rusu. Area-efficient drawings of outerplanar graphs. In G. Liotta, editor, *Graph Drawing (GD '03)*, pages 129–134, 2003.

[GR03b] A. Garg and A. Rusu. Area-efficient order-preserving planar straight-line drawings of ordered trees. *Int. J. Comput. Geometry Appl.*, 13(6):487–505, 2003.

[GR03c] A. Garg and A. Rusu. Straight-line drawings of general trees with linear area and arbitrary aspect ratio. In V. Kumar, M. L. Gavrilova, C. J. K. Tan, and P. L'Ecuyer, editors, *Computational Science and its Applications (ICCSA '03)*, volume 2669 of *LNCS*, pages 876–885, 2003.

[GR04] A. Garg and A. Rusu. Straight-line drawings of binary trees with linear area and arbitrary aspect ratio. *J. Graph Alg. Appl.*, 8(2):135–160, 2004.

[GR07] A. Garg and A. Rusu. Area-efficient planar straight-line drawings of outerplanar graphs. *Discr. Appl. Math.*, 155(9):1116–1140, 2007.

[GS69] K. R. Gabriel and R. R. Sokal. A new statistical approach to geographic variation analysis. *Systematic Zoology*, 18:259–270, 1969.

[GT94] A. Garg and R. Tamassia. Efficient computation of planar straight-line upward drawings. In *Graph Drawing (GD '94)*, pages 298–306, 1994.

[GT01] A. Garg and R. Tamassia. On the computational complexity of upward and rectilinear planarity testing. *SIAM J. Comput.*, 31(2):601–625, 2001.

[GT02] M. T. Goodrich and R. Tamassia. *Algorithm Design.* John Wiley and Sons, New York, NY, 2002.

[Har72] F. Harary. *Graph Theory.* Addison-Wesley, Reading, MA, 1972.

[He97] X. He. Grid embedding of 4-connected plane graphs. *Discr. Computat. Geom.*, 17(3):339–358, 1997.

[HL96] M. D. Hutton and A. Lubiw. Upward planarity testing of single-source acyclic digraphs. *SIAM J. Comput.*, 25(2):291–311, 1996.

[HP66] F. Harary and G. Prins. The block-cutpoint-tree of a graph. *Publ. Math. Debr.*, 13:103–107, 1966.

[HT74] J. Hopcroft and R. E. Tarjan. Efficient planarity testing. *J. ACM*, 21(4):549–568, 1974.

[JJKL08] V. Jelinek, E. Jelinkova, J. Kratochvil, and B. Lidicky. Clustered planarity: Embedded clustered graphs with two-component clusters. In M. Patrignani and I. Tollis, editors, *Graph Drawing (GD '08)*, LNCS, pages 121–132, 2008.

[JKK$^+$07] E. Jelinkova, J. Kara, J. Kratochvil, M. Pergel, O. Suchy, and T. Vyskocil. Clustered planarity: Small clusters in eulerian graphs. In S. H. Hong, T. Nishizeki, and W. Quan, editors, *Graph Drawing (GD '07)*, volume 4875 of *LNCS*, pages 303–314, 2007.

[Kan96] G. Kant. Drawing planar graphs using the canonical ordering. *Algorithmica*, 16(1):4–32, 1996.

[Kan97] G. Kant. A more compact visibility representation. *Int. J. Comput. Geom. Appl.*, 7(3):197–210, 1997.

[Kau07] M. Kaufmann. Polynomial area bounds for MST embeddings of trees. In S. H. Hong, T. Nishizeki, and W. Quan, editors, *Graph Drawing (GD '07)*, volume 4875 of *LNCS*, pages 88–100, 2007.

[KB97] G. Kant and H. L. Bodlaender. Triangulating planar graphs while minimizing the maximum degree. *Inf. Comput.*, 135(1):1–14, 1997.

[KH93] G. Kant and X. He. Two algorithms for finding rectangular duals of planar graphs. In J. van Leeuwen, editor, *Graph-Theoretic Concepts in Computer Science (WG '93)*, volume 790 of *LNCS*, pages 396–410, 1993.

[Kim95] S.K. Kim. Simple algorithms for orthogonal upward drawings of binary and ternary trees sung. In *Canadian Conference on Computational Geometry (CCCG '95)*, pages 115–120, 1995.

[Kin06] J. A. King. Realization of degree 10 minimum spanning trees in 3-space. In *Canadian Conference on Computational Geometry (CCCG '06)*, 2006.

[Kir88] D. G. Kirkpatrick. Establishing order in planar subdivisions. *Discr. Computat. Geom.*, 3:267–280, 1988.

[KKM29] B. Knaster, C. Kuratowski, and C. Mazurkiewicz. Ein beweis des fixpunktsatzes fur n dimensionale simplexe. *Fund. Math.*, 14:132–137, 1929.

[Kle07] R. Kleinberg. Geographic routing using hyperbolic space. In *INFOCOM '07*, pages 1902–1909, 2007.

[Kur30] K. Kuratowski. Sur le problme des courbes gauches en topologie. *Fund. Math.*, 15:271–283, 1930.

[KW01] M. Kaufmann and D. Wagner, editors. *Drawing Graphs, Methods and Models*, volume 2025 of *LNCS*. Spinger, 2001.

[LB95] G. Liotta and G. Di Battista. Computing proximity drawings of trees in the 3-dimensional space. In S. G. Akl, F. K. H. A. Dehne, J.-R. Sack, and N. Santoro, editors, *Algorithms and Data Structures (WADS '95)*, pages 239–250, 1995.

[Lee56] J. Leech. The problem of the thirteen spheres. *Math. Gaz.*, 40:22–23, 1956.

[Lio95] G. Liotta. *Computing Proximity Drawings of Graphs*. PhD thesis, University of Rome "La Sapienza", Italy, 1995.

[LL96] W. Lenhart and G. Liotta. Proximity drawings of outerplanar graphs. In S. C. North, editor, *Graph Drawing (GD '96)*, volume 1190 of *LNCS*, pages 286–302, 1996.

[LM08] T. Leighton and A. Moitra. Some results on greedy embeddings in metric spaces. In *Foundations of Computer Science (FOCS '08)*, pages 337–346, 2008.

[LS93] A. Lubiw and N. Sleumer. Maximal outerplanar graphs are relative neighborhood graphs. In *Canadian Conference on Computational Geometry (CCCG '93)*, pages 198–203, 1993.

[LTTV97] G. Liotta, R. Tamassia, I. G. Tollis, and P. Vocca. Area requirement of gabriel drawings. In G. Bongiovanni, D. P. Bovet, and G. Di Battista, editors, *Italian Conference on Algorithms and Complexity (CIAC '97)*, volume 1203 of *LNCS*, pages 135–146, 1997.

[Mit99] J. S. B. Mitchell. Guillotine subdivisions approximate polygonal subdivisions: A simple polynomial-time approximation scheme for geometric TSP, k-MST, and related problems. *SIAM J. Comput.*, 28(4):1298–1309, 1999.

[MNN01] Kazuyuki Miura, Shin-Ichi Nakano, and Takao Nishizeki. Grid drawings of 4-connected plane graphs. *Discr. Computat. Geom.*, 26(1):73–87, 2001.

[MS92] C. L. Monma and S. Suri. Transitions in geometric minimum spanning trees. *Discr. Computat. Geom.*, 8:265–293, 1992.

[NC88] T. Nishizeki and N. Chiba. *Planar Graphs: Theory and Algorithms*. North-Holland, Amsterdam, 1988.

[NR04] T. Nishizeki and M. S. Rahman. *Planar Graph Drawing*. Word Scientific, Singapore, 2004.

[Pap94] A. Papakostas. Upward planarity testing of outerplanar dags. In R. Tamassia and I. G. Tollis, editors, *Graph Drawing (GD '94)*, volume 894 of *LNCS*, pages 298–306, 1994.

[PCJ97] H. C. Purchase, R. F. Cohen, and M. I. James. An experimental study of the basis for graph drawing algorithms. *ACM J. Exp. Alg.*, 2:4, 1997.

[PR05] C. H. Papadimitriou and D. Ratajczak. On a conjecture related to geometric routing. *Theor. Comput. Sci.*, 344(1):3–14, 2005.

[PS85] F. Preparata and M. I. Shamos. *Computational Geometry: An Introduction*. Springer, New York, NY, 1985.

[Pur00] H. C. Purchase. Effective information visualisation: a study of graph drawing aesthetics and algorithms. *Interacting with Computers*, 13(2):147–162, 2000.

[PV84] C. H. Papadimitriou and U. V. Vazirani. On two geometric problems related to the traveling salesman problem. *J. Algorithms*, 5:231–246, 1984.

[PV04] P. Penna and P. Vocca. Proximity drawings in polynomial area and volume. *Comput. Geom. Th. Appl.*, 29(2):91–116, 2004.

[Rab93] S. Rabinowitz. $O(n^3)$ bounds for the area of a convex lattice n-gon. *Geocombinatorics*, 2:85–88, 1993.

[Rot05] G. Rote. Strictly convex drawings of planar graphs. In *Symposium on Discrete Algorithms (SODA '05)*, pages 728–734, 2005.

[RPSS03] A. Rao, C. H. Papadimitriou, S. Shenker, and I. Stoica. Geographic routing without location information. In D. B. Johnson, A. D. Joseph, and N. H. Vaidya, editors, *Conference on Mobile Computing and Networking (MOBICOM 2003)*, pages 96–108, 2003.

[RT86] P. Rosenstiehl and R. E. Tarjan. Rectilinear planar layouts and bipolar orientations of planar graphs. *Discr. Computat. Geom.*, 1:343–353, 1986.

[Sch90] W. Schnyder. Embedding planar graphs on the grid. In *Symposium on Discrete Algorithms (SODA '90)*, pages 138–148, 1990.

[Shi76] Y. Shiloach. *Arrangements of Planar Graphs on the Planar Lattice*. PhD thesis, Weizmann Institute for Science, 1976.

[SJTV08] O. Suchy, V. Jelinek, M. Tesar, and T. Vyskocil. Clustered planarity: Clusters with few outgoing edges. In M. Patrignani and I. Tollis, editors, *Graph Drawing (GD '08)*, LNCS, pages 102–113, 2008.

[SKC00] C.S. Shin, S. K. Kim, and K.Y. Chwa. Area-efficient algorithms for straight-line tree drawings. *Comput. Geom. Th. Appl.*, 15(4):175–202, 2000.

[ST92] W. Schnyder and W. Trotter. Convex drawings of planar graphs. *Abstracts of the AMS*, 92T-05-135, 1992.

[Ste51] S. K. Stein. Convex maps. *Amer. Math. Soc.*, 2:464–466, 1951.

[Sud04] M. Suderman. Pathwidth and layered drawings of trees. *Int. J. Comp. Geom. Appl.*, 14(3):203–225, 2004.

[Tam87] R. Tamassia. On embedding a graph in the grid with the minimum number of bends. *SIAM J. Comput.*, 16(3):421–444, 1987.

[Tho80] C. Thomassen. Planarity and duality of finite and infinite graphs. *J. Comb. Theory, Ser. B*, 29(2):244–271, 1980.

[Tho84] C. Thomassen. Plane representations of graphs. In *Progress in Graph Theory*, pages 43–69. Academic Press, 1984.

[TT86] R. Tamassia and I. G. Tollis. A unified approach a visibility representation of planar graphs. *Discr. Computat. Geom.*, 1:321–341, 1986.

[Tut60] W. T. Tutte. Convex representations of graphs. *London Math. Soc.*, 10:304–320, 1960.

[Tut63] W. T. Tutte. How to draw a graph. *London Math. Soc.*, 13(3):743–768, 1963.

[Val81] L. G. Valiant. Universality considerations in VLSI circuits. *IEEE Trans. Comp.*, 30(2):135–140, 1981.

[VTL82] J. Valdes, R. E. Tarjan, and E. L. Lawler. The recognition of series parallel digraphs. *SIAM J. Comput.*, 11(2):298–313, 1982.

[Wag36] K. Wagner. Bemerkungen zum vierfarbenproblem. *Jahresbericht. German. Math.-Verein*, 2:26–32, 1936.

[Wag37] K. Wagner. Uber eine eigenschaft der ebenen komplexe. *Math. Ann.*, 114:570–590, 1937.

[Woo82] D. Woods. *Drawing Planar Graphs.* PhD thesis, Stanford University, CA, 1982.

[ZH03] H. Zhang and X. He. Compact visibility representation and straight-line grid embedding of plane graphs. In F. K. H. A. Dehne, J. R. Sack, and M. H. M. Smid, editors, *Algorithms and Data Structures (WADS '03)*, volume 2748 of *LNCS*, pages 493–504, 2003.