Algorithmica (2018) 80:2286–2323
https://doi.org/10.1007/s00453-017-0322-4

# The Partial Visibility Representation Extension Problem

**Steven Chaplick**[1] · **Grzegorz Guśpiel**[2] · **Grzegorz Gutowski**[2] · **Tomasz Krawczyk**[2] · **Giuseppe Liotta**[3]

Received: 19 August 2016 / Accepted: 10 May 2017 / Published online: 24 May 2017

**Abstract** For a graph $G$, a function $\psi$ is called a *bar visibility representation* of $G$ when for each vertex $v \in V(G)$, $\psi(v)$ is a horizontal line segment (*bar*) and $uv \in E(G)$ if and only if there is an unobstructed, vertical, $\varepsilon$-wide line of sight between $\psi(u)$ and $\psi(v)$. Graphs admitting such representations are well understood (via simple characterizations) and recognizable in linear time. For a directed graph $G$, a

The conference version of this paper appeared in GD 2016 [10].

This work was partially supported by ESF Project EUROGIGA GraDR and preliminary ideas were formed during HOMONOLO 2014. Grzegorz Guśpiel was partially supported by the MNiSW Grant DI2013 000443. Grzegorz Gutowski was partially supported by the Polish National Science Center Grant UMO-2011/03/D/ST6/01370. Tomasz Krawczyk was partially supported by the Polish National Science Center Grant UMO-2015/17/B/ST6/01873. Giuseppe Liotta was partially supported by the MIUR project AMANDA "Algorithmics for MAssive and Networked DAta", prot. 2012C4E3KT_001.

✉ Grzegorz Gutowski
gutowski@tcs.uj.edu.pl

Steven Chaplick
steven.chaplick@uni-wuerzburg.de

Grzegorz Guśpiel
guspiel@tcs.uj.edu.pl

Tomasz Krawczyk
krawczyk@tcs.uj.edu.pl

Giuseppe Liotta
giuseppe.liotta@unipg.it

1 Lehrstuhl für Informatik I, Universität Würzburg, Würzburg, Germany

2 Theoretical Computer Science Department, Faculty of Mathematics and Computer Science, Jagiellonian University, Kraków, Poland

3 Universitá degli Studi di Perugia, Perugia, Italy

bar visibility representation of $G$, additionally, puts the bar $\psi(u)$ strictly below the bar $\psi(v)$ for each directed edge $(u, v)$ of $G$. We study a generalization of the recognition problem where a function $\psi'$ defined on a subset $V'$ of $V(G)$ is given and the question is whether there is a bar visibility representation $\psi$ of $G$ with $\psi(v) = \psi'(v)$ for every $v \in V'$. We show that for undirected graphs this problem, and other closely related problems, is NP-complete, but for certain cases involving directed graphs it is solvable in polynomial time.

**Keywords** Partial representation extension · Planar graphs · Bar visibility · SPQR-trees · St-ordering · NP-completeness

## 1 Introduction

The concept of a visibility representation of a graph is a classic one in computational geometry and graph drawing and the first studies on this concept date back to the early days of these fields (see, e.g. [29,47,49] for a recent survey). In the most general setting, a visibility representation of a graph is defined as a collection of disjoint sets from an Euclidean space such that the vertices are bijectively mapped to the sets and the edges correspond to unobstructed lines of sight between two such sets. Many different classes of visibility representations have been studied via restricting the space (e.g., to be the plane), the sets (e.g., to be points [6] or line segments [7,47]) and/or the lines of sight (e.g., to be non-crossing or axis-parallel). In this work we focus on a classic visibility representation setting in which the sets are horizontal line segments (*bars*) in the plane and the lines of sight are vertical. As such, whenever we refer to a visibility representation, we mean one of this type. The study of such representations was inspired by the problems in VLSI design [45,46] and was conducted by different authors [24,40,43] under variations of the notion of visibility. Tamassia and Tollis [47] gave an elegant unification of different definitions and we follow their approach.

A *horizontal bar* is an open, non-degenerate segment parallel to the $x$-axis of the coordinate plane. For a set $\Gamma$ of pairwise disjoint horizontal bars, a *visibility ray* between two bars $a$ and $b$ in $\Gamma$ is a vertical closed segment spanned between bars $a$ and $b$ that intersects $a$, $b$, and no other bar in $\Gamma$. A *visibility gap* between two bars $a$ and $b$ in $\Gamma$ is an axis aligned, non-degenerate open rectangle spanned between bars $a$ and $b$ that intersects no other bar.

For a graph $G$, a *visibility representation* $\psi$ is a function that assigns a distinct horizontal bar to each vertex such that these bars are pairwise disjoint and satisfy additional visibility constraints. Following Tamassia and Tollis [47], we distinguish three different visibility models:

- *Weak visibility* In this model, for each edge $\{u, v\}$ of $G$, there is a visibility ray between $\psi(u)$ and $\psi(v)$ in $\psi(V(G))$.
- *Strong visibility* In this model, two vertices $u, v$ of $G$ are adjacent if and only if there is a visibility ray between $\psi(u)$ and $\psi(v)$ in $\psi(V(G))$.
- *Bar visibility* In this model, two vertices $u, v$ of $G$ are adjacent if and only if there is a visibility gap between $\psi(u)$ and $\psi(v)$ in $\psi(V(G))$.

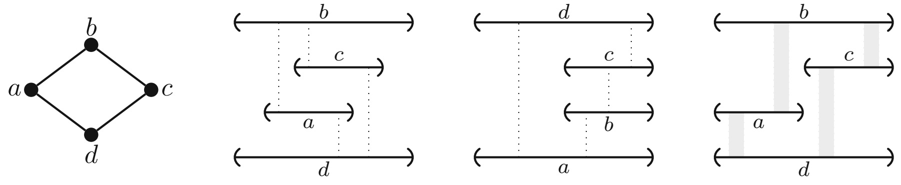

**Fig. 1** Representation of the cycle $C_4$ in weak, strong, and bar visibility model

The bar visibility model is also known as the $\varepsilon$-visibility model in the literature. See Fig. 1 for an example that shows different representations of the cycle $C_4$ in three visibility models.

A graph that admits a visibility representation in any of these models is a planar graph, but the converse does not hold in general. Tamassia and Tollis [47] characterized the graphs that admit a visibility representation in these models as follows. A graph admits a weak visibility representation if and only if it is planar. A graph admits a bar visibility representation if and only if it has a planar embedding with all cut-vertices on the boundary of the outer face. For both of these models, Tamassia and Tollis [47] presented linear-time algorithms for the recognition of representable graphs, and for constructing the appropriate visibility representations. The situation is different for the strong visibility model. Although the planar graphs admitting a strong visibility representation are characterized in [47] (via strong $st$-numberings), Andreae [1] proved that the recognition of these graphs is NP-complete. Summing up, from a computational point of view, the problems of recognizing graphs that admit visibility representations and of constructing such representations are well understood.

Recently, a lot of attention has been paid to the question of extending partial representations of graphs. In this setting a representation of some vertices of the graph is already fixed and the task is to find a representation of the whole graph that extends the given partial representation (see, e.g. [5,9,35–38] for papers that study computational aspects of extending partial representations of geometric intersection graphs). Problems of this kind are often encountered in graph drawing and are sometimes computationally harder than testing for existence of an unconstrained drawing. The problem of extending partial drawings of planar graphs is a good illustration of this phenomenon. On the one hand, by Fáry's theorem [25], every planar graph can be drawn in the plane so that each vertex is represented as a point, and edges are pairwise non-crossing, straight-line segments joining the corresponding points. Moreover, such a drawing can be constructed in linear time [11,21,22]. On the other hand, testing whether a partial drawing of this kind (i.e., an assignment of points to some of the vertices) can be extended to a straight-line drawing of the whole graph is NP-hard [44]. However, an analogous problem in the model that allows the edges to be drawn as arbitrary curves instead of straight-line segments has a linear-time solution [2]. A similar phenomenon occurs when we consider contact representations of planar graphs. Every planar graph is representable as a disk contact graph [39] or a triangle contact graph [20]. Every bipartite planar graph is representable as a contact graph of horizontal and vertical segments in the plane [19,31]. Although such representations can be constructed in polynomial time [19,20,41], the problems of extending partial representations of these kinds are NP-hard [8].

In this paper we initiate the study of extending partial visibility representations of graphs. From a practical point of view, it may be worth recalling that visibility representations are not only an appealing way of drawing graphs, but they are also typically used as an intermediate step towards constructing visualizations of networks in which all edges are oriented in a common direction and some vertices are aligned (for example to highlight critical activities in a PERT diagram). Visibility representations are also used to construct orthogonal drawings with at most two bends per edge. See, e.g. [12] for more details about these applications of visibility representations. The partial representation extension problem that we study in this paper occurs, for example, when we want to use visibility representations to incrementally draw a large network and we want to preserve the user's mental map in a visual exploration that adds a few vertices and edges per time.

Both for weak visibility and for strong visibility, the partial representation extension problems are easily found to be NP-hard. For weak visibility, the hardness follows easily from results on contact representations by Chaplick et al. [8]. For strong visibility, it follows trivially from results by Andreae [1]. Our contribution is the study of the partial representation extension problem for bar visibility representations. Hence, the central problem for this paper is the following:

Bar Visibility Representation Extension:

**Input:** $(G, \psi')$, where $G$ is a graph and $\psi'$ is a mapping assigning bars to some subset $V'$ of $V(G)$.

**Question:** Does $G$ admit a bar visibility representation $\psi$ with $\psi|V' = \psi'$?

In Sect. 5 we show that this problem is NP-complete.

**Theorem 1** *The Bar Visibility Representation Extension problem is* NP*-complete.*

The proof is a standard reduction from PlanarMonotone3Sat problem, which is known to be NP-complete thanks to de Berg and Khosravi [18]. The reduction uses gadgets that simulate logic gates and constructs a planar Boolean circuit that encodes the given formula.

We investigate a few natural modifications of the problem. Most notably, we provide some efficient algorithms for extension problems for directed graphs. A visibility representation introduces a natural orientation to edges of the graph—each edge is oriented from the lower bar to the upper one. The function $\psi$ is a representation of a digraph $G$ if, additionally to satisfying visibility constraints, it puts the bar $\psi(u)$ strictly below the bar $\psi(v)$ for each directed edge $(u, v)$ of $G$. Note that a planar digraph that admits a visibility representation also admits an *upward planar drawing* (see e.g., [27]), that is, a drawing in which the edges are represented as non-crossing monotonic upward curves.

A *source* (*sink*) of a digraph is a vertex without incoming (outgoing) edges. A *planar st-graph* is a planar acyclic digraph with exactly one source $s$ and exactly one sink $t$ that admits a planar embedding such that $s$ and $t$ are on the outer face. Di

Battista and Tamassia [14] proved that the following three statements are equivalent for a planar digraph $G$:

- $G$ admits an upward planar drawing,
- $G$ is a subgraph of a planar $st$-graph,
- $G$ admits a weak visibility representation.

Garg and Tamassia [28] showed that the recognition of planar digraphs that admit an upward planar drawing is NP-complete. It follows that the recognition of planar digraphs that admit a weak visibility representation is NP-complete, and so is the corresponding partial representation extension problem. Nevertheless, the situation might be different for bar visibility. In Sect. 3 we prove the following lemma that characterizes planar digraphs that admit a bar visibility representation.

**Lemma 2** *Let $st(G)$ be a graph constructed from a planar digraph $G$ by adding two vertices $s$ and $t$, the edge $(s, t)$, an edge $(s, v)$ for each source vertex $v$ of $G$, and an edge $(v, t)$ for each sink vertex $v$ of $G$. A planar directed graph $G$ admits a bar visibility representation if and only if the graph $st(G)$ is a planar $st$-graph.*

Since planar $st$-graphs can be recognized in linear time, planar digraphs that admit a bar visibility representation are also recognizable in linear time. The natural problem that arises is the following:
Bar Visibility Representation Extension for Digraphs:

**Input:** $(G, \psi')$, where $G$ is a directed graph and $\psi'$ is a mapping assigning bars to some subset $V'$ of $V(G)$.

**Question:** Does $G$ admit a bar visibility representation $\psi$ with $\psi|V' = \psi'$?
Although we do not provide a solution for this problem, we present an efficient algorithm for an important variant. A bar visibility representation $\psi$ of a directed graph $G$ is called *rectangular* if $\psi$ has a unique bar $\psi(s)$ with the lowest $y$-coordinate, a unique bar $\psi(t)$ with the highest $y$-coordinate, $\psi(s)$ and $\psi(t)$ span the same $x$-interval, and all other bars are inside the rectangle spanned between $\psi(s)$ and $\psi(t)$. See Fig. 2 for an example of a rectangular bar visibility representation of a planar $st$-graph.

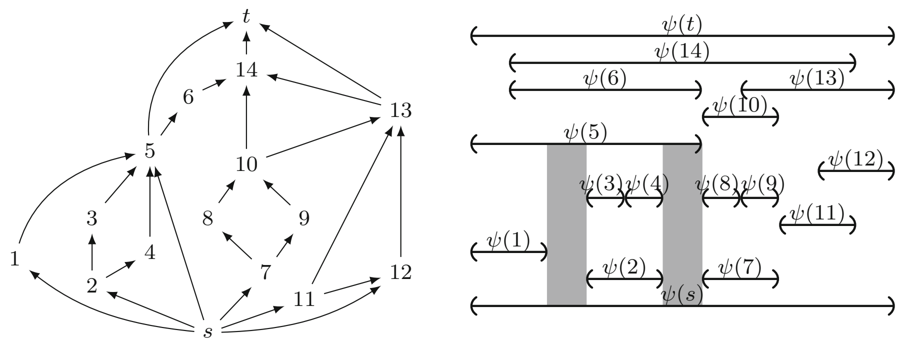

**Fig. 2** A planar $st$-graph $G$ and a *rectangular bar* visibility representation $\psi$ of $G$

Tamassia and Tollis [47] showed that a planar digraph $G$ admits a rectangular bar visibility representation if and only if $G$ is a planar $st$-graph. In Sect. 4 we give an efficient algorithm for the following problem:
Rectangular Bar Visibility Representation Extension for planar $st$-graphs:

**Input:** $(G, \psi')$, where $G$ is a planar $st$-graph and $\psi'$ is a mapping assigning bars to some subset $V'$ of $V(G)$.

**Question:** Does $G$ admit a rectangular bar visibility representation $\psi$ with $\psi|V' = \psi'$?
The main result in this paper, presented in Sect. 4, is the following.

**Theorem 3** *The rectangular bar visibility representation extension problem for a planar $st$-graph with $n$ vertices can be solved in $O\big(n \log^2 n\big)$ time.*

Our algorithm exploits the correspondence between bar visibility representations and $st$-orientations of planar graphs, and utilizes the *SPQR*-decomposition of planar graphs.

In the study of planar graphs and their representations, it is important to understand the area requirements of a drawing. A common way to measure this is by the smallest integer grid in which a drawing can be realized (see, e.g. [4,32–34,48] for papers that specifically study the area required by visibility representations of graphs and Di Battista and Frati [13] for a survey on compact drawings of graphs).

A visibility representation is a *grid representation* when all bars used in the representation have integral coordinates. Any visibility representation can be easily modified into a grid representation. However, this transformation does not preserve coordinates of the bars. In particular, it might not preserve the partial representation. We can show that the (Rectangular) Bar Visibility Representation Extension problem is NP-hard on series-parallel planar $st$-graphs when one demands a grid representation.

Our results use different tools developed for graph representation problems. In particular, we exploit the correspondence between bar visibility representations and $st$-orientations of planar graphs, and utilize the *SPQR*-decomposition for planar graphs.

The rest of the paper is organized as follows. Section 2 contains the necessary definitions and description of the necessary tools. Section 3 contains a characterization of planar digraphs that admit a bar visibility representation. Section 4 contains the study of rectangular representations of planar $st$-graphs. Section 5 contains hardness results for grid representations and for the bar visibility representation extension problem for undirected graphs. Section 6 contains conclusions and some open problems.

## 2 Preliminaries

### 2.1 Notation

A *horizontal bar* is an open, non-degenerate segment parallel to the $x$-axis of the coordinate plane. For a horizontal bar $a$, functions $y(a)$, $l(a)$, $r(a)$ give the $y$-coordinate of $a$, the $x$-coordinate of the left end of $a$, and the $x$-coordinate of the right end of $a$ respectively. For any bounded object $Q$ in the plane, we use functions $X(Q)$ and $Y(Q)$ to denote the smallest possible, possibly degenerate, closed interval containing the projection of $Q$ on the $x$-, and on the $y$-axis respectively. We denote the left end of $X(Q)$ by $l(Q)$ and the right end of $X(Q)$ by $r(Q)$. Let $a$ and $b$ be two horizontal bars with $y(a) < y(b)$. We say that $Q$ is *spanned between $a$ and $b$* if $X(Q) \subseteq X(a) \cap X(b)$, and $Y(Q) = [y(a), y(b)]$.

For a graph $G$, a visibility representation $\psi$ in any model (see Sect. 1) is a function that assigns distinct, pairwise disjoint horizontal bars to the vertices of $G$. We often describe the representation $\psi$ by providing the values of functions $y_\psi = y(\psi(v))$, $l_\psi = l(\psi(v))$, $r_\psi = r(\psi(v))$ for any vertex $v$ of $G$. We drop the subscripts when the representation $\psi$ is known from the context.

### 2.2 Planar $st$-Graphs and Their Properties

Given a graph $G = (V, E)$, a *planar drawing* of $G$ is a geometric representation of $G$ in the plane such that: $(i)$ each vertex $v \in V$ is drawn as a distinct point $p_v$; $(ii)$ each edge $e = (u, v) \in E$ is drawn as a simple curve connecting $p_u$ and $p_v$; $(iii)$ no two edges intersect except at their common end-vertices (if they are adjacent). A graph is *planar* if it admits a planar drawing.

Let $G$ be a connected planar graph. A planar drawing $\Theta$ of $G$ divides the plane into topologically connected regions, called *faces*. Exactly one face of $\Theta$ is an infinite region, and is called the *external face* of $\Theta$; the other faces are called *internal*. Each internal face is described by the counter-clockwise sequence of vertices and edges that form its boundary; the external face is described by the clockwise sequence of vertices and edges of its boundary. The description of the set of (internal and external) faces determined by a planar drawing of $G$ is called a *planar embedding* of $G$.

Let $G$ be a planar $st$-graph. An *$st$-embedding* of $G$ is any planar embedding with $s$ and $t$ on the boundary of the outer face. A planar $st$-graph together with an $st$-embedding is called a *plane $st$-graph*. Vertices $s$ and $t$ of a planar (plane) $st$-graph are called the *poles* of $G$. We abuse notation and we use the term *planar (plane) $uv$-graph* to mean a planar (plane) $st$-graph with poles $u$ and $v$. An *inner vertex* of $G$ is a vertex of $G$ other than the poles of $G$. A real valued function $\xi$ from $V(G)$ is an *$st$-valuation* of $G$ if for each edge $(u, v)$ we have $\xi(u) < \xi(v)$.

Tamassia and Tollis [47] showed that the following properties are satisfied for any plane $st$-graph:

(1) For every inner face $f$, the boundary of $f$ consists of two directed paths with a common origin and a common destination.
(2) The boundary of the outer face consists of two directed paths, with a common origin $s$ and a common destination $t$.
(3) For every inner vertex $v$, its incoming (outgoing) edges are consecutive around $v$.

To illustrate the above properties, observe in Fig. 2 two paths on the boundary of the face $(s, 2, 3, 5, 1)$, and the alignment of incoming and outgoing edges around vertex 5.

Let $G$ be a plane $st$-graph. We introduce two special objects associated with the outer face of $G$: the *left outer face $s^*$* and the *right outer face $t^*$*. Let $e = (u, v)$ be an edge of $G$. The *left face* (*right face*) of $e$ is the face of $G$ that is to the left (right) of $e$

when we traverse $e$ from $u$ to $v$. If the outer face of $G$ is to the left (right) of $e$ then we say that the left face (right face) of $e$ is $s^*$ ($t^*$).

Using property (1) we can define the *left path* and the *right path* for each inner face of $G$ as follows. If $f$ is an inner face of $G$ then the left path (right path) of $f$ consists of edges from the boundary of $f$ for which $f$ is the right face (left face). For example, in Fig. 2, the path $(s, 1, 5)$ is the left path of the face $(s, 2, 3, 5, 1)$ and the path $(s, 2, 3, 5)$ is the right path of this face.

Using property (2) we can define the left path for $t^*$ and the right path for $s^*$ as follows. The right path of $s^*$ consists of edges from the boundary of the outer face that have the outer face on their left side. The left path of $t^*$ consists of edges from the boundary of the outer face that have the outer face on their right side. The left path for $s^*$ and the right path for $t^*$ are not defined. For example, in Fig. 2, the path $(s, 1, 5, t)$ is the right path of $s^*$ and the path $(s, 12, 13, t)$ is the left path of $t^*$.

Using property (3) we can define the *left face* and the *right face* for each vertex of $G$ as follows. The left face (right face) of an inner vertex $v$ is the unique face $f$ incident to $v$ such that there are two edges $e_1$ and $e_2$ on the right path (left path) of $f$, where $e_1$ is an incoming edge for $v$ and $e_2$ is an outgoing edge for $v$. If the left face (right face) of $v$ is the outer face of $G$, we say that the left face (right face) of $u$ is $s^*$ ($t^*$). We also say that $s^*$ ($t^*$) is the left face (right face) of $s$ and $t$. For example, in Fig. 2, the left face of vertex 5 is $s^*$ and the right face of this vertex is the face $(s, 7, 8, 10, 14, 6, 5)$.

Let $G$ be a plane $st$-graph. Let $F$ be the set of inner faces of $G$ together with $s^*$ and $t^*$. The *dual* of $G$ is the directed graph $G^*$ with vertex set $F$ and edge set consisting of all pairs $(f, g)$ such that there exists an edge $e$ of $G$ with $f$ being the left face of $e$ and $g$ being the right face of $e$. Di Battista and Tamassia [14] showed that $G^*$ is a planar $s^*t^*$-graph.

Let $G$ be a plane $st$-graph and let $G^*$ be the dual of $G$. For two faces $f$ and $g$ in $V(G^*)$ we say that $f$ *is to the left of* $g$, and that $g$ *is to the right of* $f$, if there is a directed path from $f$ to $g$ in $G^*$.

### 2.3 *SPQR*-Trees for Planar $st$-Graphs

An *SPQR-tree* for a planar graph $G$ is usually used to describe all possible planar embeddings of $G$. In this paper we employ a specific version of *SPQR*-trees that allows us to describe all $st$-embeddings of a planar $st$-graph. Di Battista and Tamassia [15] were the first to define such *SPQR*-trees, and to prove the properties presented in this section.

Let $G$ be a planar $st$-graph. A *cut-vertex* of $G$ is a vertex whose removal disconnects $G$. A *separation pair* of $G$ is a pair of vertices whose removal disconnects $G$. A *split pair* of $G$ is either a separation pair or a pair of adjacent vertices. A *split component* of a split pair $\{u, v\}$ is either an edge $(u, v)$ or a maximal subgraph $C$ of $G$ such that $C$ is a planar $uv$-graph and $\{u, v\}$ is not a split pair of $C$. A *maximal split pair* $\{u, v\}$ of $G$ is a split pair such that there is no other split pair $\{u', v'\}$ where $\{u, v\}$ is contained in some split component of $\{u', v'\}$.

An *SPQR-tree* $T$ for a planar $st$-graph $G$ is a recursive decomposition of $G$ with respect to the split pairs of $G$. The tree $T$ is rooted and its nodes are of four types:

$S$ for *series nodes*, $P$ for *parallel nodes*, $Q$ for *edge nodes*, and $R$ for *rigid nodes*. Each node $\mu$ of $T$ represents a planar $st$-graph (a subgraph of $G$) called the *pertinent digraph* of $\mu$ and denoted by $G_\mu$. We use $s_\mu$ and $t_\mu$ to denote the poles of $G_\mu$: $s_\mu$ is the source of $G_\mu$, and $t_\mu$ is the sink of $G_\mu$. The pertinent digraph of the root node of $T$ is $G$. Each node $\mu$ of $T$ has an associated directed multigraph $skel(\mu)$ called the *skeleton* of $\mu$. If $\mu$ is not the root of the tree, then let $\lambda$ be the parent of $\mu$ in $T$. The node $\mu$ is associated with an edge of the skeleton of $\lambda$, called the *virtual edge* of $\mu$, which connects the poles of $G_\mu$ and represents $G_\mu$ in $skel(\lambda)$. The tree $T$ is defined recursively as follows.

- *Trivial case* If $G$ consists of a single edge $(s, t)$, then $T$ is simply a $Q$-node $\mu$. The skeleton $skel(\mu)$ is $G$.
- *Series case* If $G$ is a chain of biconnected components $G_1, \dots, G_k$ for some $k \geqslant 2$ and $c_1, \dots, c_{k-1}$ are the cut-vertices encountered in this order on any path from $s$ to $t$, then the root of $T$ is an $S$-node $\mu$ with children $\mu_1, \dots, \mu_k$. Let $c_0 = s$ and $c_k = t$. The skeleton $skel(\mu)$ is the directed path $c_0, \dots, c_k$. The pertinent digraph of $\mu_i$ is $G_i$, and edge $(c_{i-1}, c_i)$ of $skel(\mu)$ is the virtual edge of $\mu_i$.
- *Parallel case* If $\{s, t\}$ is a split pair of $G$ with split components $G_1, \dots, G_k$ for some $k \geqslant 2$, then the root of $T$ is a $P$-node $\mu$ with children $\mu_1, \dots, \mu_k$. The skeleton $skel(\mu)$ has $k$ parallel edges $(s, t)$: $e_1, \dots, e_k$. The pertinent digraph of $\mu_i$ is $G_i$, and edge $e_i$ of $skel(\mu)$ is the virtual edge of $\mu_i$.
- *Rigid case* If none of the above applies, let $\{s_1, t_1\}, \dots, \{s_k, t_k\}$ for some $k \geqslant 2$ be the maximal split pairs of $G$. For $i = 1, \dots, k$, let $G_i$ be the union of all split components of $\{s_i, t_i\}$. The root of $T$ is an $R$-node $\mu$ with children $\mu_1, \dots, \mu_k$. The skeleton $skel(\mu)$ is obtained from $G$ by replacing each subgraph $G_i$ with an edge $e_i = (s_i, t_i)$. The pertinent digraph of $\mu_i$ is $G_i$, and edge $e_i$ of $skel(\mu)$ is the virtual edge of $\mu_i$.

Note also that there is no additional edge between the poles of the skeleton of a series, parallel or rigid node—this is the only difference in the *SPQR*-tree definition given above and the one given in [15]. In particular, our definition ensures that we have a one-to-one correspondence between the edges of $skel(\mu)$ and the children of $\mu$. See Fig. 3 for an example of an *SPQR*-decomposition of the planar $st$-graph presented in Fig. 2.

Observe that the skeleton of a rigid node has only two $st$-embeddings, one being the flip of the other around the poles of the node. The skeleton of a parallel node with $k$ children has $k!$ $st$-embeddings, one for every permutation of the edges of $skel(\mu)$. The skeleton of a series node or a edge node has only one $st$-embedding.

There is a correspondence between $st$-embeddings of a planar $st$-graph $G$ and $st$-embeddings of the skeletons of $P$-nodes and $R$-nodes in the *SPQR*-tree $T$ for $G$. Having selected an $st$-embedding of the skeleton of all $P$-nodes and all $R$-nodes, we can construct an embedding of $G$ as follows. Let $t$ be the root of $T$. We replace every virtual edge $(u, v)$ in the embedding of $skel(t)$ with the recursively defined embedding of the pertinent digraph of a child of $t$ associated with the edge $(u, v)$. On the other hand, any $st$-embedding of $G$ determines:

- one of the two possible flips of the skeleton of every $R$-node in $T$;
- a permutation of the edges in the skeleton of every $P$-node.

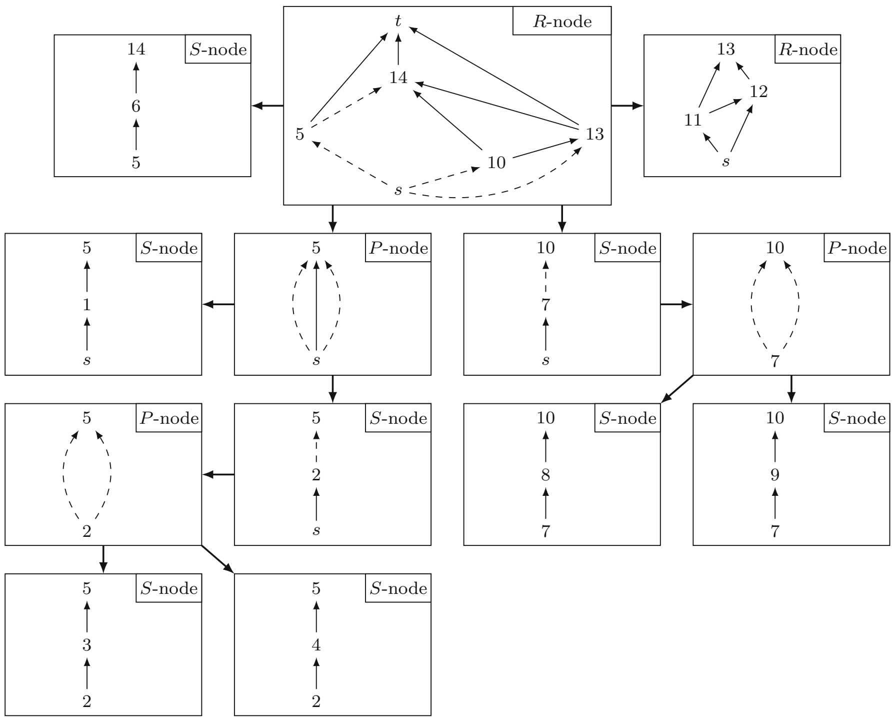

**Fig. 3** The *SPQR*-tree for the graph in Fig. 2. The $Q$-nodes (leaves of the tree) have been omitted for clarity. For each $S$-, $P$-, and $R$-node, the skeleton is given such that *each solid edge* corresponds to a $Q$-node child and *each dashed edge* corresponds to a $S$-, $P$-, or $R$-node child

Di Battista and Tamassia [15] showed that the *SPQR*-tree $T$ for a planar $st$-graph with $n$ vertices has $O(n)$ nodes, and that the total number of edges of all skeletons is $O(n)$. Gutwenger and Mutzel [30] showed that the *SPQR*-tree can be computed in linear time.

## 2.4 NP-Complete Problems

Our hardness proofs use reductions from the following NP-complete problems:

3PARTITION:

**Input:** A set of positive integers $w, a_1, a_2, \ldots, a_{3m}$ such that for each $i = 1, \ldots, 3m$, we have $\frac{w}{4} < a_i < \frac{w}{2}$.

**Question:** Can $\{a_1, \ldots, a_{3m}\}$ be partitioned into $m$ triples, such that the total sum of each triple is exactly $w$?

3PARTITION is known to be strongly NP-complete [26], i.e., the problem remains NP-complete even when the integers given in the input are encoded in unary.

PLANARMONOTONE3SAT:

**Input:** A *rectilinear* planar representation of a 3SAT formula in which each variable is a horizontal segment on the $x$-axis, each clause is a horizontal segment above or

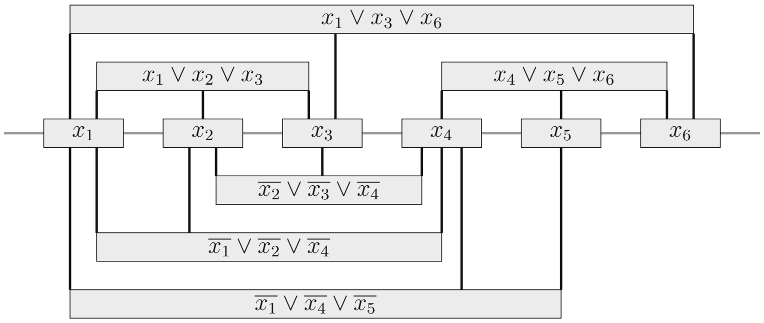
$x_1 \vee x_3 \vee x_6$
$x_1 \vee x_2 \vee x_3$
$x_4 \vee x_5 \vee x_6$
$x_1$
$x_2$
$x_3$
$x_4$
$x_5$
$x_6$
$\overline{x_2} \vee \overline{x_3} \vee \overline{x_4}$
$\overline{x_1} \vee \overline{x_2} \vee \overline{x_4}$
$\overline{x_1} \vee \overline{x_4} \vee \overline{x_5}$

**Fig. 4** A PLANARMONOTONE3SAT formula with variables $x_1, \ldots, x_6$; positive clauses $\{x_1, x_3, x_6\}$, $\{x_1, x_2, x_3\}$, and $\{x_4, x_5, x_6\}$; and negative clauses $\{\overline{x_1}, \overline{x_4}, \overline{x_5}\}$, $\{\overline{x_1}, \overline{x_2}, \overline{x_4}\}$, and $\{\overline{x_2}, \overline{x_3}, \overline{x_4}\}$

below the $x$-axis with straight-line vertical connections to the variables it includes. All positive clauses are above the $x$-axis and all negative clauses are below the $x$-axis. There are no clauses including both positive and negative occurrences of variables, all horizontal segments are pairwise disjoint, and each vertical connection intersects only with the two segments that it connects. See Fig. 4 for an example.

**Question:** Is the formula satisfiable?

PLANARMONOTONE3SAT is known to be NP-complete thanks to de Berg and Khosravi [18].

## 3 Bar Visibility and Rectangular Bar Visibility Representations for Planar Digraphs

A bar visibility representation $\psi$ of a planar $st$-graph is *rectangular* when $X(\psi(s)) = X(\psi(t))$ and for any vertex $v$ we have $X(\psi(v)) \subseteq X(\psi(s))$. Tamassia and Tollis [47] observed the following connection between planar $st$-graphs and rectangular bar visibility representations. Any collection of pairwise disjoint bars $\Gamma$ with the bottom-most bar $s$ and the top-most bar $t$ that satisfies $X(s) = X(t)$, and $X(a) \subseteq X(s)$ for every $a \in \Gamma$, naturally induces a planar $st$-graph on the set $\Gamma$—a digraph containing all edges $(a, b)$ such that $a$ is strictly below $b$ and there is a visibility gap between $a$ and $b$ in $\Gamma$. They further showed that every planar $st$-graph has a rectangular bar visibility representation.

The next lemma characterizes the planar digraphs that admit a bar visibility representation. For a planar digraph $G$, let $st(G)$ be a graph constructed from $G$ by adding two vertices $s$ and $t$, the edge $(s, t)$, an edge $(s, v)$ for each source vertex $v$ of $G$, and an edge $(v, t)$ for each sink vertex $v$ of $G$.

**Lemma 2** *A planar directed graph $G$ admits a bar visibility representation if and only if the graph $st(G)$ is a planar $st$-graph.*

*Proof* Suppose that $st(G)$ is a planar $st$-graph. Tamassia and Tollis [47] showed that $st(G)$ has a rectangular bar visibility representation $\psi$ with the bottom-most bar $\psi(s)$ and the top-most bar $\psi(t)$. Clearly, $\psi|V(G)$ is a bar visibility representation for $G$.

Conversely, assume that $\psi$ is a bar visibility representation of $G$ and $\Gamma$ is the image of $\psi$. For every bar $a \in \Gamma$, let $A(a)$ ($B(a)$) be the interior of the set of all $x$ such that $\psi(a)$ is the first encountered bar from $\Gamma$ if we traverse downward (upward) the vertical line with the $x$-coordinate $x$. We say that a bar $a \in \Gamma$ is *visible from above (below)* if $A(a) \neq \emptyset$ ($B(a) \neq \emptyset$). Note that each $A(a)$ and $B(a)$ is a union of disjoint open intervals. Observe also that if $a$ represents a sink (a source) of $G$ then $A(a) = (l(a), r(a))$ ($B(a) = (l(a), r(a))$). Otherwise, $\psi$ would not be a bar visibility representation of $G$.

We claim that some bars in $\Gamma$ can be extended so that the new set of bars still represents $G$ and has the property that only the bars representing the sources are visible from below and only the bars representing the sinks are visible from above. Before we give a proof of this claim, suppose that $\psi$ satisfies this property. We can define two bars $\psi(s)$ and $\psi(t)$ such that $X(s) = X(t)$, $X(\bigcup \Gamma) \subsetneq X(s)$, and $y_\psi(s) < y_\psi(v) < y_\psi(t)$ for every vertex $v$ of $G$. This extension of $\psi$ is a rectangular bar representation of $st(G)$. It follows that $st(G)$ is a planar $st$-graph.

Now we show that $\psi$ can be modified so that the bars visible from below (above) represent the sources (sinks) of $G$. Let $\mathcal{X}$ denote the set of the $x$-coordinates of all end-points of all bars in $\Gamma$. Let $\varepsilon$ and $\delta$ be respectively the minimum and the maximum difference between any two values in $\mathcal{X}$. Suppose that there is a bar $\psi(v)$ in $\Gamma$ that is visible from below and $v$ is not a source of $G$. Suppose $(L, R)$ is an interval of $B(\psi(v))$ and observe that both $L$ and $R$ are in $\mathcal{X}$. Since $v$ is not a source of $G$ and $\psi$ is a visibility representation of $G$, there is a vertex $u$ in $G$ such that $(u, v)$ is an edge of $G$, and either $r_\psi(u) = L$ or $l_\psi(u) = R$. If $r_\psi(u) = L$, we extend $\psi(u)$ to the right so that $r_\psi(u) = R$, and if $l_\psi(u) = R$, we extend $\psi(u)$ to the left so that $l_\psi(u) = L$. Observe that such a modification only introduces additional visibility gaps between $\psi(u)$ and $\psi(v)$, and does not change any visibility gap between any other two bars. Thus, the modified $\psi$ remains a representation of $G$. Moreover, all end-points of all bars are still in $\mathcal{X}$ and the total length of all bars increases by at least $\varepsilon$. We repeat the same procedure as long as there is a vertex $v$ that is not a source of $G$ and $\psi(v)$ is visible from below. The number of repetitions is bounded, as the length of any single bar never exceeds $\delta$. In the resulting representation, each bar visible from below represents a source of $G$. Next, we transform $\psi$ again, using a similar algorithm, so that each bar visible from above represents a sink of $G$. It is easy to see, that in the second step we do not introduce any new bars that are visible from below. Thus, in the resulting representation, all the bars visible from below are sources of $G$ and all the bars visible from above are sinks of $G$. $\square$

Lemma 2 gives a linear-time algorithm for the recognition of planar digraphs that admit a bar visibility representation. Recall from the discussion in Sect. 1 that the recognition of planar digraphs that admit a weak visibility representation is an NP-complete problem. It follows that the extension problem for digraphs in the weak visibility model is NP-complete. We do not know the complexity status for the extension problem for digraphs neither in the strong nor in the bar visibility model. Nevertheless, the results in Sect. 4 give hope for a polynomial-time algorithm for the extension problem for bar visibility representations of digraphs.

## 4 Rectangular Bar Visibility Representations of Planar $st$-Graphs

In this section we solve the following problem.
Rectangular Bar Visibility Representation Extension for planar $st$-graphs:

**Input:** $(G, \psi')$, where $G$ is a planar $st$-graph and $\psi'$ is a mapping assigning bars to some subset $V'$ of $V(G)$.

**Question:** Does $G$ admit a rectangular bar visibility representation $\psi$ with $\psi|V' = \psi'$?

As our algorithm is rather technical (involving many small details), we now provide a high level description of the main ideas. In the first step, our algorithm calculates $y$-coordinates $y_\psi$ for our potential bars. Namely, the algorithm checks whether $y_{\psi'} : V' \to \mathbb{R}$ is extendable to an $st$-valuation $y_\psi$ of $G$. When such an extension does not exist, the algorithm rejects; otherwise it turns out (as shown in Lemma 10) that any extension of $y_{\psi'}$ can be used as $y_\psi$. To determine whether there is a set of $x$-coordinates to match this set of $y$-coordinates, we use a dynamic programming approach which proceeds bottom-up through the *SPQR*-tree $T$ of the given planar $st$-graph $G$. Recall that, as discussed in Sect. 2.3, $T$ provides a hierarchical decomposition of $G$ according to separation pairs, i.e., each node $\mu$ of $T$ corresponds to a separation pair $(u, v)$ in $G$. Additionally, for a separation pair $(u, v)$ corresponding to a node $\mu$ in $T$, the subgraph 'between' $u$ and $v$ is a planar $st$-graph, and is precisely the pertinent digraph of $\mu$. We will see that there is a similar connection between the *SPQR*-tree and each rectangular bar visibility representation $\psi$ of $G$. Namely, we describe how $\psi$ 'decomposes' into sub-representations according to these separation pairs, i.e., for each node $\mu$ and its corresponding separation pair $(u, v)$, there is a particular rectangular bar visibility representation $\psi_\mu$ of the pertinent digraph $G_\mu$ of $\mu$ in $\psi$. Moreover, we will see that each $\psi_\mu$ partitions into rectangular 'tiles' where each 'tile' is a representation of one of its children. We say that a 'tile' of a representation $\psi_\mu$ of $G_\mu$ is *valid* when it extends $\psi'|V(G_\mu)$. Now, essentially, the key to our dynamic program is to efficiently describe the set of possible valid 'tiles' of the representations of $G_\mu$ in terms of the valid 'tiles' of $\mu$'s children. It turns out that it is sufficient to consider four types of 'tiles' for each node of $T$ in order to accomplish this. To efficiently determine the types of 'tiles' admitted by a node $\mu$, we distinguish different cases depending on whether $\mu$ is a $P$-node, an $S$-node, a $Q$-node, or an $R$-node. As usual, for dynamic programming involving *SQPR*-trees, the $R$-node case is the most complex. So, we describe it first in using a quadratic-time subroutine which is more intuitive. We then describe a speed-up of this subroutine which runs in nearly linear time (this subroutine remains as the main bottleneck for our running time).

Section 4.1 presents structural properties of bar visibility representations in relation to an *SPQR*-decomposition. We describe how a rectangular bar visibility representation of the pertinent digraph of a node $\mu$ in the *SPQR*-decomposition is composed of rectangular bar visibility representations of the pertinent digraphs of the children of $\mu$. In Sect. 4.2 we present an algorithm that solves this extension problem in quadratic time. In Sect. 4.3 we refine the algorithm to work in $O\left(n \log^2 n\right)$ time for a planar $st$-graph with $n$ vertices.

## 4.1 Structural Properties

Let $\Gamma$ be a collection of pairwise disjoint bars. For a pair of bars $a$, $b$ in $\Gamma$ with $y(a) < y(b)$ let the *set of visibility rectangles* $R(a, b)$ be the interior of the set of points $(x, y)$ in $\mathbb{R}^2$ that satisfy the following properties:

(1) $a$ is the first bar in $\Gamma$ on a vertical line downwards from $(x, y)$,
(2) $b$ is the first bar in $\Gamma$ on a vertical line upwards from $(x, y)$.

Figure 2 shows (shaded area) the set of visibility rectangles $R(s, 5)$. Note that there is a visibility gap between $a$ and $b$ in $\Gamma$ if and only if $R(a, b)$ is non-empty. Additionally, if $R(a, b)$ is non-empty, then it is a union of pairwise disjoint open rectangles spanned between $a$ and $b$.

Let $G$ be a planar $st$-graph and let $T$ be the *SPQR*-tree for $G$. Let $\psi$ be a rectangular bar visibility representation of $G$. For every node $\mu$ of $T$ we define the set $B_\psi(\mu)$, called the *bounding box of* $\mu$ *with respect to* $\psi$, as the closure of the following union:

$$\bigcup \left\{ R(\psi(u), \psi(v)) : (u, v) \text{ is an edge of the pertinent digraph } G_\mu \right\}.$$

If $\psi$ is clear from the context, then the set $B_\psi(\mu)$ is denoted by $B(\mu)$ and is called the *bounding box of* $\mu$. Let $B(\psi) = X(\psi(V(G))) \times Y(\psi(V(G)))$ be the minimal closed axis-aligned rectangle that contains the representation $\psi$. It follows from the definition of rectangular embedding, and from the definition of bounding box, that:

(1) $B(\psi) = B_\psi(\mu)$, where $\mu$ is the root of $T$,
(2) each point in $B(\psi)$ is in the closure of at least one set of visibility rectangles $R(\psi(u), \psi(v))$ for some edge $(u, v)$ of $G$,
(3) each point in $B(\psi)$ is in at most one set of visibility rectangles.

The following two lemmas describe basic properties of a bounding box.

**Lemma 4** (Q-Tiling Lemma) *Let $\mu$ be a $Q$-node in $T$ that corresponds to an edge $(u, v)$ of $G$. For any rectangular bar visibility representation $\psi$ of $G$ we have:*

(1) *$B(\mu)$ is a union of pairwise disjoint rectangles spanned between $\psi(u)$ and $\psi(v)$.*
(2) *If $B(\mu)$ is not a single rectangle, then the parent $\lambda$ of $\mu$ in $T$ is a $P$-node, and $u$, $v$ are the poles of the pertinent digraph $G_\lambda$.*

*Proof* The first assertion is obvious. Suppose that $B(\mu)$ is a union of at least 2 rectangles. Let $R_1$ and $R_2$ be the two left-most rectangles of $B(\mu)$. Consider the rectangle $S$ spanned between $\psi(u)$ and $\psi(v)$ and between the right side of $R_1$ and left side of $R_2$. There are some bars in $\psi(G)$ that are contained in $S$. The vertices corresponding to these bars together with $u$ and $v$ form a planar $uv$-graph. Hence, the split pair $\{u, v\}$ has at least two split components: the edge $(u, v)$ and at least one other component. Thus, $\lambda$ is a $P$-node with poles $u$ and $v$.

In Fig. 5 observe that the set $R(s, 5)$ is a union of two rectangles. Recall the *SPQR*-decomposition presented in Fig. 3 and that the $Q$-node corresponding to the edge $(s, 5)$ is a child of a $P$-node.

The Basic Tiling Lemma presented below describes the relation between the bounding box of an inner node $\mu$ and the bounding boxes of the children of $\mu$ in any rectangular bar visibility representation of $G$. In particular, the next lemma justifies the use of the name *bounding box* for the set $B(\mu)$.

**Lemma 5** (Basic Tiling Lemma) *Let $\mu$ be an inner node in $T$ with children $\mu_1, \ldots, \mu_k$, $k \geqslant 2$. For any rectangular bar visibility representation $\psi$ of $G$ we have:*

(1) *$\psi(v) \subseteq B(\mu)$ for every inner vertex $v$ of $G_\mu$.*
(2) *$B(\mu)$ is a rectangle that is spanned between $\psi(s_\mu)$ and $\psi(t_\mu)$.*
(3) *The sets $B(\mu_1), \ldots, B(\mu_k)$ tile the rectangle $B(\mu)$, i.e., $B(\mu_1), \ldots, B(\mu_k)$ cover $B(\mu)$ and the interiors of $B(\mu_1), \ldots, B(\mu_k)$ are pairwise disjoint.*

*Proof* Observe that for an inner vertex $v$ of $G_\mu$, any edge of $G$ incident to $v$ is an edge of $G_\mu$. The closures of the sets of visibility rectangles corresponding to all edges incident to $v$ cover $\psi(v)$ and property (1) follows.

To prove (2) note that for every inner vertex $v$ of $G_\mu$, the set

$$S_\mu(v) = X(\psi(v)) \times [y(\psi(s_\mu)), y(\psi(t_\mu))]$$

is a rectangle that is spanned between $\psi(s_\mu)$ and $\psi(t_\mu)$ and it is internally disjoint from $\psi(w)$ for any vertex $w$ not in $V(G_\mu)$. Otherwise, there would be a visibility gap that would correspond to an edge between an inner vertex of $G_\mu$ and a vertex in $V(G) \setminus V(G_\mu)$.

If $(s_\mu, t_\mu)$ is not an edge of $G_\mu$, then

$$B(\mu) = \bigcup \{S_\mu(v) : v \text{ is an inner vertex of } G_\mu\};$$

otherwise

$$B(\mu) = \bigcup \{S_\mu(v) : v \text{ is an inner vertex of } G_\mu\} \cup R(\psi(s_\mu), \psi(t_\mu)).$$

In both cases $B(\mu)$ is a rectangle spanned between $\psi(s_\mu)$ and $\psi(t_\mu)$.

Property (3) follows immediately from the fact that the edges of $G_{\mu_1}, \ldots, G_{\mu_k}$ form a partition of the edges of $G_\mu$. □

In the next three lemmas we extend the Basic Tiling Lemma by a more precise description of tilings of the bounding box of an inner node $\mu$ by the bounding boxes of the children of $\mu$. We separately consider the cases that $\mu$ is a $P$-node, an $S$-node, and an $R$-node. Figures 5, and 6 give a graphical presentation of the Tiling Lemmas. In Lemmas 6, 7, and 9 we assume that $\mu_1, \ldots, \mu_k$ are the children of $\mu$ for some $k \geqslant 2$.

**Lemma 6** (P-Tiling Lemma) *Let $\mu$ be a $P$-node. For any rectangular bar visibility representation $\psi$ of $G$ we have:*

(1) *If $(s_\mu, t_\mu)$ is not an edge of $G$, then the sets $B(\mu_1), \ldots, B(\mu_k)$ are rectangles spanned between $\psi(s_\mu)$ and $\psi(t_\mu)$.*

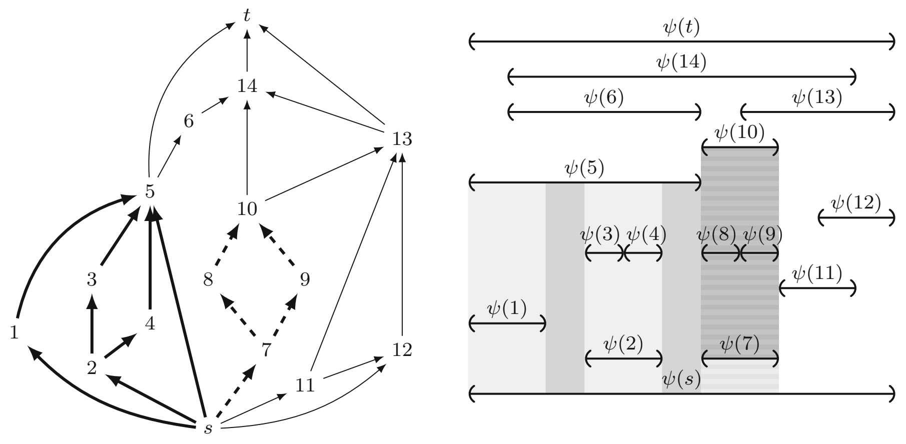

**Fig. 5** The graph $G$, the pertinent digraph of the $P$-node $\mu_1$ (*solid thick edges*), the pertinent digraph of the $S$-node $\mu_2$ (*dashed thick edges*), the representation $\psi(G)$, the tiling of $\mu_1$ in $\psi(G)$ (*solid fill*), and the tiling of $\mu_2$ in $\psi(G)$ (*patterned fill*)

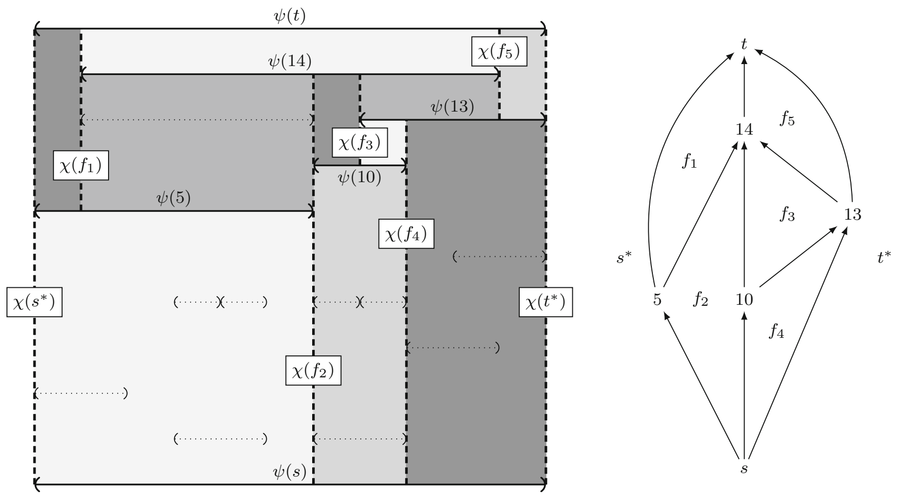

**Fig. 6** The tiling of the $R$-node $\mu$, the embedding $\mathcal{E}$ of $skel(\mu)$, splitting lines (*dashed*) for faces of $\mathcal{E}$, and the $st$-valuation $\chi$ of $\mathcal{E}^*$

(2) *If $(s_\mu, t_\mu)$ is an edge of G, then $\mu$ has exactly one child that is a Q-node, say $\mu_1$, and:*
- *For $i = 2, \dots, k$, $B(\mu_i)$ is a rectangle spanned between $\psi(s_\mu)$ and $\psi(t_\mu)$.*
- *$B(\mu_1)$ is a non-empty union of rectangles spanned between $\psi(s_\mu)$ and $\psi(t_\mu)$.*

*Proof* This is an immediate consequence of the Basic Tiling Lemma and Lemma 4. □

When $\mu$ is an $S$-node or an $R$-node, then there is no edge $(s_\mu, t_\mu)$. By the Q-Tiling Lemma and by the Basic Tiling Lemma, each set $B(\mu_i)$ is a rectangle that is spanned between the bars representing the poles of $G_{\mu_i}$.

**Lemma 7** (S-Tiling Lemma) *Let $\mu$ be an $S$-node. Let $c_1, \dots, c_{k-1}$ be the cut-vertices of $G_\mu$ encountered in this order on a path from $s_\mu$ to $t_\mu$. Let $c_0 = s_\mu$, and $c_k = t_\mu$. For any rectangular bar visibility representation $\psi$ of $G$, for every $i = 1, \dots, k-1$, we have $X(\psi(c_i)) = X(B(\mu))$. For every $i = 1, \dots, k$, $B(\mu_i)$ is spanned between $\psi(c_{i-1})$ and $\psi(c_i)$ and $X(B(\mu_i)) = X(B(\mu))$.*

*Proof* Suppose to the contrary, that the bar assigned to cut-vertex $c_i$ is the first one that does not span the whole interval $X(B(\mu))$. This creates a gap of visibility between a vertex in the $i$-th biconnected component and a vertex in one of the later components. This contradicts $c_i$ being a cut-vertex.

The R-Tiling Lemma should describe all possible tilings of the bounding box of an $R$-node $\mu$ that appear in all representations of $G$. Since there is a one-to-one correspondence between the edges of $skel(\mu)$ and the children of $\mu$, we abuse notation and write $B(u, v)$ to denote the bounding box of the child of $\mu$ that corresponds to the edge $(u, v)$ of $skel(\mu)$. By the Basic Tilling Lemma, $B(u, v)$ is spanned between the bars representing $u$ and $v$.

Suppose that $\psi$ is a representation of $G$. The tiling $\tau = (B_\psi(\mu_1), \dots, B_\psi(\mu_k))$ of $B_\psi(\mu)$ determines a triple $(\mathcal{E}, \xi, \chi)$, where:

- $\mathcal{E}$ is an $s_\mu t_\mu$-embedding of $skel(\mu)$,
- $\xi$ is an $st$-valuation of $\mathcal{E}$,
- $\chi$ is an $st$-valuation of $\mathcal{E}^*$,

that are defined as follows.

Consider the following planar drawing of the planar $st$-graph $skel(\mu)$. Draw every vertex $u$ in the middle of $\psi(u)$, and every edge $e = (u, v)$ as a curve that starts in the middle of $\psi(u)$, goes a little above $\psi(u)$ towards the rectangle $B_\psi(u, v)$, goes inside $B_\psi(u, v)$ towards $\psi(v)$, and a little below $\psi(v)$ to the middle of $\psi(v)$. This way we obtain a plane $st$-graph $\mathcal{E}$, which is an $st$-embedding of $skel(\mu)$. The $st$-valuation $\xi$ of $\mathcal{E}$ is just the restriction of $y_\psi$ to the vertices from $skel(\mu)$, i.e., $\xi = y_\psi | V(skel(\mu))$. To define the $st$-valuation $\chi$ of $\mathcal{E}^*$ we use the following lemma.

**Lemma 8** (Face Condition)

(1) *Let $f$ be a face in $V(\mathcal{E}^*)$ different than $t^*$, and $v_0, v_1, \dots, v_n$ be the right path of $f$. There is a vertical line $L_r(f)$ that contains the left endpoints of $\psi(v_1), \dots, \psi(v_{n-1})$ and the left sides of $B_\psi(v_0, v_1), \dots, B_\psi(v_{n-1}, v_n)$.*
(2) *Let $f$ be a face in $V(\mathcal{E}^*)$ different than $s^*$, and $u_0, u_1, \dots, u_m$ be the left path of $f$. There is a vertical line $L_l(f)$ that contains the right endpoints of $\psi(u_1), \dots, \psi(u_{m-1})$ and the right sides of $B_\psi(u_0, u_1), \dots, B_\psi(u_{m-1}, u_m)$.*
(3) *If $f$ is an inner face of $\mathcal{E}$ then $L_l(f) = L_r(f)$.*

*Proof* To prove (1) we first show that for every $i = 1, \dots, n-1$, we have

$$l(\psi(v_i)) = l(B_\psi(v_{i-1}, v_i)).$$

By the Basic Tiling Lemma, $B_\psi(v_{i-1}, v_i)$ is a rectangle spanned between $\psi(v_{i-1})$ and $\psi(v_i)$. It follows that $l(\psi(v_i)) \leqslant l(B_\psi(v_{i-1}, v_i))$. Suppose that $l(\psi(v_i)) < l(B_\psi(v_{i-1}, v_i))$. By the Basic Tiling Lemma again, there is a child $\lambda$ of $\mu$ such that the rectangle $B_\psi(\lambda)$ has its top right corner located at the intersection of the left side of $B_\psi(v_{i-1}, v_i)$ and $\psi(v_i)$. Clearly, $\lambda$ corresponds to an edge of $skel(\mu)$ that is in the embedding $\mathcal{E}$ between $(v_{i-1}, v_i)$ and $(v_i, v_{i+1})$ in the clockwise order around $v_i$. However, there is no such edge in $\mathcal{E}$, a contradiction. Similarly, for every $i = 1, \ldots, n-1$, we have

$$l(\psi(v_i)) = l(B_\psi(v_i, v_{i+1})).$$

It follows that the left sides of the bounding boxes $B_\psi(v_0, v_1), \ldots, B_\psi(v_{n-1}, v_n)$ and the left endpoints of $\psi(v_1), \ldots, \psi(v_{n-1})$ are aligned to the same vertical line $L_r(f)$.

The proof of (2) is analogous. Property (3) is an immediate consequence of the Basic Tiling Lemma. □

The above lemma allows us to introduce the notion of a *splitting line* for every face $f$ in $V(\mathcal{E}^*)$; namely, the splitting line of $f$ is: the line $L_l(f) = L_r(f)$ if $f$ is an inner face of $\mathcal{E}$, $L_r(f)$ if $f$ is the left outer face of $\mathcal{E}$, and $L_l(f)$ if $f$ is the right outer face of $\mathcal{E}$. Now, let $\chi(f)$ be the $x$-coordinate of the splitting line for a face $f$ in $V(\mathcal{E}^*)$. To show that $\chi(f)$ is an $st$-valuation of $\mathcal{E}^*$, note that for any edge $(f, g)$ of $\mathcal{E}^*$ there is an edge $(u, v)$ of $\mathcal{E}$ that has $f$ on the left side and $g$ on the right side. It follows that $\chi(f) = l(B_\psi(u, v)) < r(B_\psi(u, v)) = \chi(g)$, proving the claim. See Fig. 6 for an illustration.

The representation $\psi$ of $G$ determines the triple $(\mathcal{E}, \xi, \chi)$. Note that any other representation with the same tiling $\tau = (B_\psi(\mu_1), \ldots, B_\psi(\mu_k))$ of $B(\mu)$ gives the same triple. To emphasize that the triple $(\mathcal{E}, \xi, \chi)$ is determined by tiling $\tau$, we write $(\mathcal{E}_\tau, \xi_\tau, \chi_\tau)$.

Now, assume that $\mathcal{E}$ is an $st$-embedding of $skel(\mu)$, $\xi$ is an $st$-valuation of $\mathcal{E}$, and $\chi$ is an $st$-valuation of the dual of $\mathcal{E}$. Consider the function $\phi$ that assigns to every vertex $v$ of $skel(\mu)$ the bar $\phi(v)$ defined as follows:

$$\begin{aligned} y_\phi(v) &= \xi(v), \\ l_\phi(v) &= \chi(\text{left face of } v), \\ r_\phi(v) &= \chi(\text{right face of } v). \end{aligned}$$

Firstly, Tamassia and Tollis [47] showed that $\phi$ is a bar visibility representation of $skel(\mu)$ and that for $\tau = (B_\phi(\mu_1), \ldots, B_\phi(\mu_k))$, we have $(\mathcal{E}_\tau, \xi_\tau, \chi_\tau) = (\mathcal{E}, \xi, \chi)$.

Secondly, there is a representation $\psi$ of $G$ that agrees with $\tau$ on $skel(\mu)$, i.e., such that $\tau = (B_\psi(\mu_1), \ldots, B_\psi(\mu_k))$. To construct such a representation, we take any representation $\psi$ of $G$, translate and scale all bars in $\psi$ to get $B_\psi(\mu) = B_\phi(\mu)$, and represent the pertinent digraphs $G_{\mu_1}, \ldots, G_{\mu_k}$ so that the bounding box of $\mu_i$ coincides with $B_\phi(\mu_i)$ for $i = 1, \ldots, k$.

The considerations given above lead us to the following lemma.

**Lemma 9** (R-Tiling Lemma) *Let $\mu$ be an R-node. There is a one-to-one correspondence between the set*

$\mathcal{T} = \{(B_\psi(\mu_1), \ldots, B_\psi(\mu_k)) : \psi$ *is a rectangular bar visibility representation of* $G\}$

*of all possible tilings of the bounding box of* $\mu$ *by the bounding boxes of* $\mu_1, \ldots, \mu_k$ *in all representations of G and the set*

$$\mathcal{T}' = \left\{ (\mathcal{E}, \xi, \chi) : \begin{array}{l} \mathcal{E} \textit{ is an } st\textit{-embedding of skel}(\mu), \\ \xi \textit{ is an } st\textit{-valuation of } \mathcal{E}, \\ \chi \textit{ is an } st\textit{-valuation of the dual of } \mathcal{E}. \end{array} \right\}$$

### 4.2 Algorithm for Rectangular Bar Visibility Extension of Planar $st$-Graphs

Let $G$ be a planar $st$-graph with $n$ vertices and $\psi'$ be a partial representation of $G$ with the set of *fixed* vertices $V'$. We present a quadratic-time algorithm that tests if there exists a rectangular bar visibility representation $\psi$ of $G$ that extends $\psi'$. If such a representation exists, then the algorithm can construct it in the same time.

In the first step, our algorithm calculates $y_\psi$. Namely, the algorithm checks whether $y_{\psi'} : V' \to \mathbb{R}$ is extendable to an $st$-valuation of $G$. When such an extension does not exist, the algorithm rejects the instance $(G, \psi')$; otherwise any extension of $y_{\psi'}$ can be used as $y_\psi$. The correctness of this step is verified by the following lemma.

**Lemma 10** *Suppose that* $\psi$ *is a rectangular bar visibility representation of G that extends* $\psi'$.

(1) *The function* $y_\psi$ *is an* $st$*-valuation of G that extends* $y_{\psi'}$,
(2) *If* $y$ *is an* $st$*-valuation of G that extends* $y_{\psi'}$, *then the function* $\phi$ *that sends every vertex* $v$ *of G into a bar so that*

$$y_\phi(v) = y(v), \quad l_\phi(v) = l_\psi(v), \quad r_\phi(v) = r_\psi(v)$$

*is also a rectangular bar visibility representation of G that extends* $\psi'$.

*Proof* The function $y_\psi$ extends $y_{\psi'}$, because $\psi$ extends $\psi'$. It is an $st$-valuation of $G$ because for an edge $(u, v)$ of $G$, the bar of $u$ is below the bar of $v$.

For the proof of (2), observe that for each vertex $u$ of $G$ we have $X(\phi(u)) = X(\psi(u))$. We claim that for any two vertices $u$, $v$ of $G$ such that the interior of $X(\psi(u)) \cap X(\psi(v))$ is non-empty we have that $y_\psi(u) < y_\psi(v)$ if and only if $y(u) < y(v)$. For the proof of this claim, let $u$ and $v$ be vertices of $G$ such that the interior of $X(\psi(u)) \cap X(\psi(v))$ is non-empty. From the fact that $\psi(V(G))$ is a collection of pairwise disjoint bars, it follows that $y_\psi(u) \neq y_\psi(v)$. Without loss of generality assume that $y_\psi(u) < y_\psi(v)$. The non-empty interior of $X(\psi(u)) \cap X(\psi(v))$ means that there is a path from $u$ to $v$ in $G$. Hence $y(u) < y(v)$ as $y$ is an $st$-valuation of $G$.

As a consequence we have that $(x_1, x_2) \times (y_\psi(u), y_\psi(v))$ is a visibility gap between bars $\psi(u)$ and $\psi(v)$ in representation $\psi$ if and only if $(x_1, x_2) \times (y(u), y(v))$ is a visibility gap between $\phi(u)$ and $\phi(v)$ and $\phi$ is a rectangular bar visibility representation of $G$. □

Clearly, checking whether $y_{\psi'}$ is extendable to an $st$-valuation of $G$, and constructing such an extension can be done in $O(n)$-time. In the second step, the algorithm computes the *SPQR*-tree $T$ for $G$, which also takes linear time.

Before we describe the last step in our algorithm, we need some preparation. For an inner node $\mu$ in $T$ we define the sets $V'(\mu)$ and $C(\mu)$ as follows:

$$V'(\mu) = \text{the set of fixed vertices in } V(G_\mu) \setminus \{s_\mu, t_\mu\},$$
$$C(\mu) = \begin{cases} \emptyset, \text{ if } V'(\mu) = \emptyset; \\ \text{the smallest axis aligned, closed rectangle that contains } \psi'(u) \text{ for} \\ \text{all } u \in V'(\mu), \text{ otherwise.} \end{cases}$$

The set $C(\mu)$ is called the *core* of $\mu$. For a node $\mu$ whose core is empty, our algorithm can represent $G_\mu$ in any rectangle spanned between the poles of $G_\mu$. Thus, we focus our attention on nodes whose core is non-empty.

Assume that $\mu$ is a node whose core is non-empty. We describe the 'possible shapes' the bounding box of $\mu$ might have in a representation of $G$ that extends $\psi'$. The bounding box of $\mu$ is a rectangle that is spanned between the bars corresponding to the poles of $G_\mu$. By the Basic Tiling Lemma, if $C(\mu)$ is non-empty then $B(\mu)$ contains $C(\mu)$. For our algorithm it is important to distinguish whether the left (right) side of $B(\mu)$ contains the left (right) side of $C(\mu)$. This criterion leads to four types of representations of $\mu$ with respect to the core of $\mu$.

The main idea of the algorithm is to decide for each inner node $\mu$ whose core is non-empty, which of the four types of representation of $\mu$ are possible and which are not. The algorithm traverses the tree bottom-up and for each node and each type of representation it tries to construct the appropriate tiling using the information about possible representations of its children. The types chosen for different children need to fit together to obtain a tiling of the parent node. In what follows, we present our approach in more detail.

Let $\mu$ be an inner node in $T$. Fix $\phi' = \psi'|V'(\mu)$. It is convenient to think of $\phi'$ as a partial representation of the pertinent digraph $G_\mu$ obtained by restricting $\psi'$ to the inner vertices of $G_\mu$. In particular, $\phi'$ is empty if the core of $\mu$ is empty. Let $x, x'$ be two real values. A rectangular bar visibility representation $\phi$ of $G_\mu$ is called an $[x, x']$-*representation of* $\mu$ if $\phi$ extends $\phi'$ and $X(\phi(s_\mu)) = X(\phi(t_\mu)) = [x, x']$.

A node $\mu$ whose core is empty has an $[x, x']$-representation for any $x < x'$. In particular, a $Q$-node has an $[x, x']$-representation for any $x < x'$.

We say that an $[x, x']$-representation of an inner node $\mu$ whose core is non-empty is:

- *left-loose, right-loose* (*LL-representation*), when $x < l(C(\mu))$ and $x' > r(C(\mu))$,
- *left-loose, right-fixed* (*LF-representation*), when $x < l(C(\mu))$ and $x' = r(C(\mu))$,
- *left-fixed, right-loose* (*FL-representation*), when $x = l(C(\mu))$ and $x' > r(C(\mu))$,
- *left-fixed, right-fixed* (*FF-representation*), when $x = l(C(\mu))$ and $x' = r(C(\mu))$.

The next lemma justifies this categorization of representations. It says that if a representation of a given type exists, then every representation of the same type is also realizable.

**Lemma 11** (Stretching Lemma) *Let $\mu$ be an inner node whose core is non-empty. We have that:*

- *If $\mu$ has an LL-representation, then $\mu$ has an $[x, x']$-representation for any $x < l(C(\mu))$ and any $x' > r(C(\mu))$.*
- *If $\mu$ has an LF-representation, then $\mu$ has an $[x, x']$-representation for any $x < l(C(\mu))$ and $x' = r(C(\mu))$.*
- *If $\mu$ has an FL-representation, then $\mu$ has an $[x, x']$-representation for $x = l(C(\mu))$ and any $x' > r(C(\mu))$.*

*Proof* Let $x_l = l(C(\mu))$. Suppose that $\phi$ is some left-loose $[x_1, x']$-representation of $\mu$ with $x_1 < x_l$. For any $x_2 < x_l$ we can obtain an $[x_2, x']$-representation of $\mu$ by appropriately stretching the part of the drawing of $\phi$ that is to the left of $x_l$. If the representation is right-loose, then we can arbitrarily stretch the part of the drawing that is to the right of $r(C(\mu))$. □

The main task of the algorithm is to verify which representations are feasible for nodes that have non-empty cores. We assume that:

- $\mu$ is an inner node whose core is non-empty.
- $\mu_1, \dots, \mu_k$ are the children of $\mu$, $k \geqslant 2$.
- $\lambda_1, \dots, \lambda_{k'}$ are the children of $\mu$ with $C(\lambda_i) \neq \emptyset$, $0 \leqslant k' \leqslant k$.
- $\theta(\lambda_i)$ is the set of feasible types of representations for $\lambda_i$, $\theta(\lambda_i) \subseteq \{LL, LF, FL, FF\}$.

We process the tree bottom-up and assume that $\theta(\lambda_i)$ is already computed and non-empty.

Let $x$ and $x'$ be two real numbers such that $x \leqslant l(C(\mu))$ and $x' \geqslant r(C(\mu))$. We provide an algorithm that tests whether an $[x, x']$-representation of $\mu$ exists. We use it to find feasible types for $\mu$ by calling it four times with appropriate values of $x$ and $x'$. While searching for an $[x, x']$-representation of $\mu$ our algorithm tries to tile the rectangle $[x, x'] \times [y(s_\mu), y(t_\mu)]$ with $B(\mu_1), \dots, B(\mu_k)$. The tiling procedure is determined by the Tiling Lemma specific for the type of $\mu$. Note that as the core of a $Q$-node is empty, the algorithm splits into three cases: $\mu$ is an $S$-node, a $P$-node, and an $R$-node.

**Case S. $\mu$ is an $S$-node** In this case we attempt to align the left and right sides of the bounding boxes of the children of $\mu$ to $x$ and $x'$ respectively. We also must have the $x$-intervals of the bars of the cut vertices set to $[x, x']$.

**Claim 12** *There exists an $[x, x']$-representation of an S-node $\mu$ if and only if Algorithm 1 returns true.*

*Proof* Claim follows directly from the Tiling Lemma for Series Nodes and the Stretching Lemma.

**Case P. $\mu$ is a $P$-node** In this case we attempt to tile the rectangle $[x, x'] \times [y(s_\mu), y(t_\mu)]$ by placing the bounding boxes of the children of $\mu$ side by side from left to right. The order of children whose cores are non-empty is determined by the

**Algorithm 1** Algorithm for series node

```
1: for each cut-vertex $c$ in $G_\mu$ do
2:     if $c \in V'(\mu)$ and $X(\psi'(c)) \neq [x, x']$ then
3:         return False        ▷ fixed cut-vertex does not span $[x, x']$
4: for $i = 1, \dots, k'$ do
5:     if $l(C(\lambda_i)) > x$ and $r(C(\lambda_i)) < x'$ and $LL \notin \theta(\lambda_i)$ then
6:         return False        ▷ $\lambda_i$ must stretch on both sides
7:     if $l(C(\lambda_i)) > x$ and $r(C(\lambda_i)) = x'$ and $LF \notin \theta(\lambda_i)$ then
8:         return False        ▷ $\lambda_i$ must stretch only on the left side
9:     if $l(C(\lambda_i)) = x$ and $r(C(\lambda_i)) < x'$ and $FL \notin \theta(\lambda_i)$ then
10:        return False        ▷ $\lambda_i$ must stretch only on the right side
11:    if $l(C(\lambda_i)) = x$ and $r(C(\lambda_i)) = x'$ and $FF \notin \theta(\lambda_i)$ then
12:        return False        ▷ $\lambda_i$ must stretch on neither side
13: return True
```

position of those cores. We need to find enough space to place the bounding boxes of children whose cores are empty. Additionally, if $(s_\mu, t_\mu)$ is an edge of $G$, then we need to leave at least one visibility gap in the tiling for that edge. Otherwise, if $(s_\mu, t_\mu)$ is not an edge of $G$, we need to close all the gaps in the tiling. The tiling algorithm is presented as Algorithm 2.

**Algorithm 2** Algorithm for parallel node

```
1: sort $\lambda_i$'s by the value $l(C(\lambda_i))$
2: for $i = 1, \dots, k'$ do
3:     $l_i \leftarrow l(C(\lambda_i)), r_i \leftarrow r(C(\lambda_i))$        ▷ left- and right- endpoints of cores
4: $r_0 \leftarrow x, l_{k'+1} \leftarrow x'$
5: $closed \leftarrow \emptyset$        ▷ set of closed gaps
6: for $i = 0, \dots, k'$ do
7:     if $r_i > l_{i+1}$ then
8:         return False        ▷ cores overlap
9:     if $r_i = l_{i+1}$ then
10:        $closed \leftarrow closed \cup \{i\}$        ▷ $\lambda_i$ and $\lambda_{i+1}$ touch
11:        if $i > 0$ then
12:            $\theta(\lambda_i) \leftarrow \theta(\lambda_i) \setminus \{FL, LL\}$        ▷ Use right-fixed rep. of $\lambda_i$
13:        if $i < k'$ then
14:            $\theta(\lambda_{i+1}) \leftarrow \theta(\lambda_{i+1}) \setminus \{LF, LL\}$        ▷ Use left-fixed rep. of $\lambda_{i+1}$
15: if $\big(k > k'$ or $(s_\mu, t_\mu) \in E(G_\mu)\big)$ and $|closed| = k' + 1$ then
16:    return False        ▷ there is no gap
17: for $i = 1, \dots, k'$ do
18:    if $\theta(\lambda_i) = \emptyset$ then
19:        return False
20:    if $LL \in \theta(\lambda_i)$ then
21:        $closed \leftarrow closed \cup \{i - 1, i\}$        ▷ close both gaps
22:    else if $i - 1 \notin closed$ and $LF \in \theta(\lambda_i)$ then
23:        $closed \leftarrow closed \cup \{i - 1\}$        ▷ close left gap
24:    else if $FL \in \theta(\lambda_i)$ then
25:        $closed \leftarrow closed \cup \{i\}$        ▷ close right gap
26: return $(s_\mu, t_\mu) \in E(G_\mu)$ or $k - k' >= k' + 1 - |closed|$        ▷ can close all gaps
```

In line 1 the children whose cores are non-empty are sorted by the left end of the core. In lines 2 to 5 the variables $l_i, r_i$, and an empty set *closed* are initialized.

If there are $\lambda_i, \lambda_j$ such that the interior of the set $X(C(\lambda_i)) \cap X(C(\lambda_j))$ is non-empty, then we prove that there is no $[x, x']$-representation of $G_\mu$. Indeed, by the Tiling Lemma for Parallel Nodes and by $C(\lambda_i) \subseteq B(\lambda_i)$, the interior of $B(\lambda_i) \cap B(\lambda_j)$ is non-empty and hence tiling of $B(\mu)$ with $B(\mu_1), \ldots, B(\mu_k)$ is not possible. Additionally, if $r(C(\lambda_i)) = l(C(\lambda_j))$, then neither a right-loose representation of $\lambda_i$ nor a left-loose representation of $\lambda_j$ can be used. These checks and restrictions are performed by the algorithm in lines 6 to 14.

Let $Q_i = [r_i, l_{i+1}] \times [y(s_\mu), y(t_\mu)]$ for $i \in [0, k']$. We say that $Q_i$ is an *open gap* (after $\lambda_i$, before $\lambda_{i+1}$) if $Q_i$ has non-empty interior. In particular, if $x = r_0 < l_1$ ($r_{k'} < l_{k'+1} = x'$) then there is an open gap before $\lambda_1$ (after $\lambda_{k'}$). On the one hand, if there is an edge $(s_\mu, t_\mu)$ or there is at least one $\mu_i$ whose core is empty, then we need at least one open gap to construct an $[x, x']$-representation. This condition is checked by the algorithm in line 15. On the other hand, if $(s_\mu, t_\mu)$ is not an edge of $G$ then we need to close all the gaps in the tiling. There are two ways to close the gaps. Firstly, the representation of each child node whose core is empty can be placed so that it closes a gap. The second way is to use loose representations for children nodes $\lambda_1, \ldots, \lambda_{k'}$.

If $\theta(\lambda_i) = \emptyset$ for some $i = 1, \ldots, k'$, then an $[x, x']$-representation of $\mu$ does not exist. Assume that $\theta(\lambda_i)$ is non-empty for every $i = 1, \ldots, k'$. Suppose that $c$ is a function that assigns to every $\lambda_i$ a feasible type of representation from the set $\theta(\lambda_i)$. Whenever $c(\lambda_i)$ is right-loose or $c(\lambda_{i+1})$ is left-loose, we can stretch the representation of $\lambda_i$ or $\lambda_{i+1}$, so that it closes the gap $Q_i$. In lines 17 to 25, the algorithm greedily closes as many gaps as possible. The greedy strategy processes children $\lambda_i$'s from left to right and for each child: closes both adjacent gaps if it can; it prefers closing the left gap if it is not yet closed (this is the last bounding box that can close this gap) from the right gap.

If there are some open gaps left and $(s_\mu, t_\mu)$ is not an edge of $G$, then each open gap needs to be closed by placing in this gap a representation of one or more of the children whose core is empty.

**Claim 13** *There exists an $[x, x']$-representation of a $P$-node $\mu$ if and only if Algorithm 2 returns true.*

*Proof* Claim follows by the Tiling Lemma for Parallel Nodes and the Stretching Lemma. The correctness of the greedy strategy follows by a simple greedy exchange argument.

**Case R. $\mu$ is an $R$-node** By the Tiling Lemma for Rigid Nodes, the set of possible tilings of $B(\mu)$ by $B(\mu_1), \ldots, B(\mu_k)$ is in correspondence with the triples $(\mathcal{E}, \xi, \chi)$, where $\mathcal{E}$ is a planar embedding of $skel(\mu)$, $\xi$ is an $st$-valuation of $\mathcal{E}$, and $\chi$ is an $st$-valuation of $\mathcal{E}^*$. To find an appropriate tiling of $B(\mu)$ (that yields an $[x, x']$-representation of $\mu$) we search through the set of such triples. Since $\mu$ is a rigid node, there are only two planar $st$-embeddings of $skel(\mu)$ and we consider both of them separately. Let $\mathcal{E}$ be one of these planar embeddings. Since the $y$-coordinate for each vertex of $G$ is already fixed, the $st$-valuation $\xi$ is given by the $y$-coordinates of the vertices from $skel(\mu)$. Now, it remains to find an $st$-valuation $\chi$ of $\mathcal{E}^*$, i.e., to determine the $x$-coordinate of the splitting line for every face $f$ of $\mathcal{E}$. First, for every face $f$ in $V(\mathcal{E}^*)$ we compute an initial set of possible placements for the splitting line of $f$ by taking into account the partial representation $\phi'$. If $f$ is an inner face of $\mathcal{E}$, then we have the following restrictions on $\chi(f)$:

- If $u$ is a fixed vertex on the left path of $f$, then $\chi(f) = r(u)$.
- If $u$ is a fixed vertex on the right path of $f$, then $\chi(f) = l(u)$.
- If $\lambda$ is a child of $\mu$ whose core is non-empty, and the virtual edge of $\lambda$ is on the left path of $f$, then $\chi(f) \geqslant r(C(\lambda))$.
- If $\lambda$ is a child of $\mu$ whose core is non-empty, and the virtual edge of $\lambda$ is on the right path of $f$, then $\chi(f) \leqslant l(C(\lambda))$.

We impose analogous conditions for the faces $s^*$ and $t^*$.

Let $\mathcal{X}'(f)$ be a set of all $\chi(f)$ in $[x, x']$ that satisfy all the above conditions. If $\mathcal{X}'(f) = \emptyset$ for some face $f$ in $V(\mathcal{E}^*)$ or $x \notin \mathcal{X}'(s^*)$ or $x' \notin \mathcal{X}'(t^*)$, then there is no $[x, x']$-representation of $G_\mu$. Since we are looking for an $[x, x']$-representation of $G_\mu$, we set $\mathcal{X}'(s^*) = [x, x]$ and $\mathcal{X}'(t^*) = [x', x']$ as the splitting line for $s^*$ ($t^*$) must be set to $x$ ($x'$).

Now, we further restrict the possible values for $\chi(f)$ by taking into account the fact that $\chi$ needs to be an $st$-valuation of $\mathcal{E}^*$. For every two faces $f$ and $g$ in $V(\mathcal{E}^*)$:

- If $g$ is to the left of $f$, then $\chi(f) > l(\mathcal{X}'(g))$.
- If $g$ is to the right of $f$, then $\chi(f) < r(\mathcal{X}'(g))$.

Let $\mathcal{X}(f)$ be the set of all $\chi(f)$ such that $\chi(f) \in \mathcal{X}'(f)$ and that satisfy the above conditions. If $\mathcal{X}(f)$ is empty for some face $f$ in $V(\mathcal{E}^*)$, then there is no $[x, x']$-representation of $G_\mu$. We assume that $\mathcal{X}(f)$ is non-empty for every $f$ in $V(\mathcal{E}^*)$. One can easily verify the following claim.

**Claim 14** (1) *For every face $f$ in $V(\mathcal{E}^*)$, $\mathcal{X}(f)$ is an interval in* $[x, x']$,
(2) *For every two faces $f$ and $g$ such that $f$ is to the left of $g$, we have that:*

- *$l(\mathcal{X}(f)) \leqslant l(\mathcal{X}(g))$ and if $l(\mathcal{X}(f)) = l(\mathcal{X}(g))$ then $\mathcal{X}(g)$ is open from the left side.*
- *$r(\mathcal{X}(f)) \leqslant r(\mathcal{X}(g))$ and if $r(\mathcal{X}(f)) = r(\mathcal{X}(g))$ then $\mathcal{X}(f)$ is open from the right side.*

A face $f$ in $V(\mathcal{E}^*)$ is *determined* if $\mathcal{X}(f)$ is a singleton (i.e., the location of the splitting line of $f$ is already fixed); otherwise $f$ is *undetermined.*

In what follows, we construct a 2-CNF formula $\Phi$ that is satisfiable if and only if an $[x, x']$-representation of $\mu$ exists.

**Variables of** $\Phi$ For every child $\lambda$ of $\mu$ whose core is non-empty, we introduce two Boolean variables: $l_\lambda$ and $r_\lambda$, which have the following interpretation:

- The true (false) value of variable $l_\lambda$ means that we use left-loose (left-fixed) representation of node $\lambda$.
- The true (false) value of variable $r_\lambda$ means that we use right-loose (right-fixed) representation of node $\lambda$.

For every inner face $f$ of $\mathcal{E}$ we introduce two Boolean variables: $l_f$ and $r_f$, which have the following interpretation:

- The variable $l_f$ is true when the splitting line of $f$ is set strictly to the right of $l(\mathcal{X}(f))$. It is false when $\chi(f) = l(\mathcal{X}(f))$.
- The variable $r_f$ is true when the splitting line of $f$ is set strictly to the left of $r(\mathcal{X}(f))$. It is false when $\chi(f) = r(\mathcal{X}(f))$.

In particular, if $\mathcal{X}(f)$ is open from the left (right) then $l_f$ ($r_f$) is true. When $f$ is determined then both $l_f$ and $r_f$ are false.

For the left outer face $s^*$ and the right outer face $t^*$ we introduce variables $r_{s^*}$ and $l_{t^*}$. Since $s^*$ and $t^*$ are determined, the corresponding variables are always set to false.

**Clauses of** $\Phi$ We split the clauses of $\Phi$ into four types.

Type I. Clauses of this type propagate the information about the possible representation types of the children of $\mu$.

- For every child $\lambda$ with non-empty core and for every type of representation of $\lambda$ which is not feasible, we add a clause that forbids using representation of this type. For example, when there is no left-loose, right-fixed representation of $\lambda$ we add a clause $\neg(l_\lambda \wedge \neg r_\lambda) = (\neg l_\lambda \vee r_\lambda)$ to $\Phi$.

Type II. Clauses of this type enforce the meaning of variables $l_f$ and $r_f$ for every face $f$ in $V(\mathcal{E}^*)$.

- For every determined inner face $f$, we add the clauses $(\neg l_f)$ and $(\neg r_f)$, and for $s^*$ and $t^*$ we add the clauses $(\neg r_{s^*})$ and $(\neg l_{t^*})$.
- For every undetermined inner face $f$, we add the clause $(l_f)$ if $\mathcal{X}(f)$ is open from the left side, and $(r_f)$ if $\mathcal{X}(f)$ is open from the right side. If $\mathcal{X}(f)$ is closed from the left and closed from the right side, we add the clause $(l_f \vee r_f)$ as the splitting line of $f$ cannot be placed in both endpoints of $\mathcal{X}(f)$ simultaneously.

Type III. Clauses of this type enforce a 'proper tiling' of every face $f$ in $V(\mathcal{E}^*)$ (see Face Condition Lemma). This ensures that the bounding boxes associated with the left path and the right path of $f$ can be aligned to the splitting line of $f$.

- For every face $f$ and for every node $\lambda$ on the left path of $f$ with a non-empty core:
  - We add the clause $(l_f \Rightarrow r_\lambda)$.
  - If $r(C(\lambda)) < l(\mathcal{X}(f))$, we add the clause $(r_\lambda)$.
  - If $r(C(\lambda)) = l(\mathcal{X}(f))$, we add the clause $(\neg l_f \Rightarrow \neg r_\lambda)$.

The clause $(l_f \Rightarrow r_\lambda)$ asserts that whenever the splitting line of $f$ is set to the right of $l(\mathcal{X}(f))$, then a right-loose representation of $\lambda$ is necessary to align the bounding box of $\lambda$ to the splitting line of $f$. The two remaining clauses have similar meaning.

We add analogous clauses for the nodes whose cores are non-empty and that correspond to the edges from the right path of $f$.

Type IV. Clauses of this type enforce that the $x$-coordinates of the splitting lines form an $st$-valuation of $\mathcal{E}^*$.

- For every pair of faces $f$ and $g$ in $V(\mathcal{E}^*)$ such that $f$ is to the left of $g$ and such that $r(\mathcal{X}(f)) \geqslant l(\mathcal{X}(g))$, we add the clause $(\neg r_f \Rightarrow l_g)$.

Such a clause forbids setting $\chi(f) = r(\mathcal{X}(f))$ and $\chi(g) = l(\mathcal{X}(g))$ – such an assignment of $\chi(f)$ and $\chi(g)$ would not be a valid $st$-valuation.

**Claim 15** *Let $\mu$ be an R-node, $\mathcal{E}$ be a planar embedding of the skeleton of $\mu$. There exists an $[x, x']$-representation of $\mu$ that corresponds to a planar embedding $\mathcal{E}$ if and only if $\Phi$ is satisfiable.*

*Proof* Suppose that $\phi$ is an $[x, x']$-representation of $\mu$. For every face $f$ in $V(\mathcal{E}^*)$, the splitting line $\chi(f)$ of $f$ in $\phi$ satisfies $\chi(f) \in \mathcal{X}(f)$. Thus, $\chi(f)$ determines the following assignment for $l_f$ and $r_f$:

- $l_f$ is true if and only if $\chi(f) > l(\mathcal{X}(f))$,
- $r_f$ is true if and only if $\chi(f) < r(\mathcal{X}(f))$.

For every child $\lambda$ of $\mu$ such that $C(\lambda) \neq \emptyset$ we set the variables $l_\lambda$ and $r_\lambda$ as follows:

- $l_\lambda$ is true if and only if $\lambda$ is left-loose in $\phi$,
- $r_\lambda$ is true if and only if $\lambda$ is right-loose in $\phi$.

One can easily check that this assignment satisfies $\Phi$.

Suppose now that $\Phi$ is satisfiable. We define an $[x, x']$-representation $\phi$ of $\mu$ by setting a splitting line $\chi(f)$ for every face $f$ in $V(\mathcal{E}^*)$. To conclude that $\phi$ is an $[x, x']$-representation it is enough to check that:

(1) The function $\chi$ is an $st$-valuation of $\mathcal{E}^*$.
(2) For every fixed vertex $u$ we have that

$$l_\phi(u) = \chi(\text{left face of } u) \text{ and } r_\phi(u) = \chi(\text{right face of } u).$$

(3) For every child $\lambda$ of $\mu$ such that $C(\lambda) \neq \emptyset$ we have that

$$G_\lambda \text{ has an } [\chi(\text{left face of } \lambda), \chi(\text{right face of } \lambda)]\text{-representation.}$$

First, we define $\chi(f) = l(\mathcal{X}(f))$ when $l_f$ is false. We also set $\chi(f) = r(\mathcal{X}(f))$ when $r_f$ is false. Note that this definition is unambiguous as both $l_f$ and $r_f$ are false only for a determined face $f$ by the fact that the clauses of Type II are satisfied. By Claim 14 and by the fact that the clauses of Type IV are satisfied, for any two faces $f$ and $g$ in $V(\mathcal{E}^*)$ for which $\chi(f)$ and $\chi(g)$ are already fixed, we have $\chi(f) < \chi(g)$ whenever $f$ is to the left of $g$. Notice that $\chi(f)$ is not yet determined for inner faces $f$ for which $l_f$ and $r_f$ are true. For such a face $f$, let $\mathcal{X}''(f)$ contain all values $z$ such that:

- $\chi(g) < z$ whenever $g$ is a face to the left of $f$ and the value $\chi(g)$ is already fixed, and
- $z < \chi(h)$ whenever $h$ is a face to the right of $f$ and the value $\chi(h)$ is already fixed, and
- $l(\mathcal{X}(f)) < z < r(\mathcal{X}(f))$.

We claim that $\mathcal{X}''(f)$ is an open, non-empty interval. Indeed, if $\mathcal{X}''(f)$ is empty, then there are faces $g$ and $h$ with the values $\chi(g)$, $\chi(h)$ fixed such that $g$ is to the left of $h$ and $\chi(g) \geqslant \chi(h)$, which contradicts our previous observation that if a face is to the left of another face and splitting lines are determined for both of them, then the splitting line of the first face is to the left of the splitting line of the second face.

Moreover, for any two faces $f_1$, $f_2$ such that $f_1$ is to the left of $f_2$ and neither $\chi(f_1)$ nor $\chi(f_2)$ is fixed, we have that $l(\mathcal{X}''(f_1)) \leqslant l(\mathcal{X}''(f_2))$ and $r(\mathcal{X}''(f_1)) \leqslant r(\mathcal{X}''(f_2))$. Thus, for every face $f$ for which both $l_f$, $r_f$ are true, we can choose a value $\chi(f)$ from $\mathcal{X}''(f)$ so that $\chi$ is an $st$-valuation of $\mathcal{E}^*$. We need to check the remaining conditions (2) and (3). Condition (2) is satisfied since, for a determined face $f$ we have chosen $\chi(f)$ from the singleton $\mathcal{X}(f)$. Condition (3) follows from the fact that the clauses of Type I and Type III are satisfied, and by the Stretching Lemma. □

### *4.2.1 Complexity Considerations*

To compute the feasible representation for a node $\mu$ with $k$ children, our algorithm works in $O(k)$-time if $\mu$ is an $S$-node. Algorithm 2 for a $P$-node $\mu$ needs to sort the children of $\mu$ and thus, it works in $O(k \log k)$-time. For an $R$-node, the number of clauses of Types I, II and III is $O(k)$. The number of clauses of Type IV is $O\big(k^2\big)$ and for some graphs, it is quadratic. Thus, the algorithm works in $O\big(k^2\big)$ time for an $R$-node. Since the number of all edges in all nodes of $T$ is $O(n)$, the whole algorithm works in $O\big(n^2\big)$ time.

## 4.3 Faster Algorithm

The bottleneck of the algorithm presented in Sect. 4.2 is the number of clauses of Type IV in the 2-CNF formula constructed for $R$-nodes. In the presented algorithm we add one clause $(\neg r_f \Rightarrow l_g)$ for any two faces $f$ and $g$ in $V(\mathcal{E}^*)$ such that $f$ is to the left of $g$ and $r(\mathcal{X}(f)) \geqslant l(\mathcal{X}(g))$. The number of such pairs of faces can be quadratic. In this section we present a different, less direct, approach that uses a smaller number of clauses to express the same set of constraints.

We can treat the planar $st$-graph $\mathcal{E}^*$ as a planar poset with a single minimal and a single maximal element. Using the result by Baker et al. [3] we know that such a poset has dimension at most 2. Thus, there are two numberings $p$ and $q$ of the vertices of $\mathcal{E}^*$ such that a face $f$ is to the left of a face $g$ if and only if $p(f) < p(g)$ and $q(f) < q(g)$. Such numberings correspond to dominance drawings of planar $st$-graphs and can be computed in linear time [16].

For each face $f$, we have two Boolean variables $l_f$ and $r_f$, two real values $\lambda_f = l(\mathcal{X}(f))$, $\varrho_f = r(\mathcal{X}(f))$, and two integer values $p_f = p(f)$ and $q_f = q(f)$. We want to introduce a small set of 2-CNF clauses that implies $(\neg l_g \Rightarrow r_f)$ whenever $p_f < p_g, q_f < q_g$, and $\varrho_f \geqslant \lambda_g$.

To give an intuition for our approach, consider the simpler problem of determining for every face $f$ the number of faces $g$ such that $p_f < p_g, q_f < q_g$, and $\varrho_f \geqslant \lambda_g$. This is a three-dimensional range counting query problem and can be solved in $O\big(n \log^3 n\big)$ time using range trees, as described in Chapter 5 of [17].

It can also be solved in $O\big(n \log^2 n\big)$ time using the following sweep line algorithm. First, we setup a dynamic data structure for two-dimensional range counting queries. Then, we process faces, one by one, in order of increasing values of $p$. After processing each face $h$, we add the point $(q_h, \varrho_h)$ to the structure. To compute the answer for a face $h$, we ask the structure for the number of points $(x, y)$ such that $x < p_h$ and $y \geqslant \lambda_h$.

Now we give an overview of our approach to the original problem of creating a small set of 2-CNF clauses. Our algorithm is a sweep line algorithm that resembles the one described above. In particular, we use a data structure similar to a two-dimensional range tree, that is simply a set of *persistent* balanced binary search trees, each with $\varrho_f$ as the sorting key. During the course of the sweep, we create additional Boolean variables corresponding to the vertices of the trees and implications corresponding to their edges. The algorithm executes $O(n)$ queries against the structure, each of the queries takes $O\big(\log^2 n\big)$ time. As a result, both the maximal size of the structure and the total running time amount to $O\big(n \log^2 n\big)$. The algorithm produces $O\big(n \log^2 n\big)$ 2-CNF clauses that correspond to the edges of the search trees and the control flow of the queries.

For an overview of persistent data structures, refer to [23]. However, we only need the ideas presented in [42], which are summarized in this paragraph. The tree structure used in our algorithm is a modification of the AVL tree. A node $\alpha$ of the tree stores a pointer to its left child $left(\alpha)$, a pointer to its right child $right(\alpha)$ and the sorting key $key(\alpha)$. The parent links are purposefully not stored – AVL trees can easily be implemented without them and we must not store them for the persistent approach to work. The difference from regular AVL trees is that no node is ever modified. Let $A$ be the set of all vertices that the insertion procedure would modify, together with all of their ancestors. It is a known property of AVL trees that $|A|$ is logarithmic in the number of vertices of the tree. In a persistent AVL tree, instead of modifying nodes in $A$, we perform the modifications on their copies – we create a new node $C(\alpha)$ for every $\alpha \in A$ and set $key(C(\alpha)) = key(\alpha)$.

This way, each addition to the tree introduces a logarithmic number of new nodes. After each addition, we get a new root node, that represents the new tree, the old tree is represented by the previous root and all but a logarithmic number of the nodes are shared by both trees. The graph of old and new nodes together with edges from nodes to their children is an acyclic digraph.

Now, in our algorithm, each tree keeps some set of Boolean variables $r_f$ sorted by the value of $\varrho_f$. More specifically, with every vertex $\alpha$ we associate a Boolean variable $var(\alpha)$. To add a variable $r_f$ into the tree we add a new node $\alpha$ with $key(\alpha) = \varrho_f$ and $var(\alpha) = r_f$. Additionally, with each node $\alpha$ of the tree we associate a second Boolean variable $var'(\alpha)$ and for each child node $\beta$ we add a clause $(var'(\alpha) \Rightarrow var'(\beta))$. Finally, for each node $\alpha$ we add a clause $(var'(\alpha) \Rightarrow var(\alpha))$.

To simplify the presentation, assume that we have $n = 2^k$ faces. Let numberings $p$ and $q$ take values $0, \dots, n-1$. We construct the 2-CNF formula in the following way. First, for each interval of integers $[j \cdot 2^i, (j+1) \cdot 2^i)$, $0 \leqslant i \leqslant k$, $j < \frac{n}{2^i}$ we have one persistent balanced binary search tree. The tree for the interval $[a, b)$ is going to keep variables $r_f$ for faces $f$ such that $q_f \in [a, b)$.

We process faces, one by one, in order of increasing values of $p$. After processing each face $f$, we add the variable $r_f$ to all trees $[a, b)$ such that $q_f \in [a, b)$. There are $k + 1$ such trees and an addition to each tree takes $O(\log n)$ time and introduces $O(\log n)$ new Boolean variables.

This way, when we process face $g$, then any earlier processed face $f$ satisfies $p_f < p_g$ and no other face satisfies this condition. Now, for any value $q_g$ we can select a logarithmic size subset $S$ of the trees such that the union of the intervals of the trees is exactly $[0, q_g)$. The variables $r_f$ stored in these trees are exactly the variables for faces that satisfy both $p_f < p_g$ and $q_f < q_g$. This is exactly the set of faces that are to the left of the face $g$.

Each tree in $S$ stores variables $r_f$ sorted by $\varrho_f$. We can execute a binary search for the left-most node with the key no smaller than $\lambda_g$. During the search, when we descend from an inner node $\alpha$ to the left child or when $\alpha$ is the final node of the search, we add clauses $(\neg l_g \Rightarrow var'(right(\alpha))$ and $(\neg l_g \Rightarrow var(\alpha))$. The first implication is forwarded over the tree to all nodes in the right subtree. This way, after completing the search, we have that $\neg l_g$ implies $r_f$ for all faces $f$ such that $p_f < p_g, q_f < q_g$ and $\varrho_f \geqslant \lambda_g$ – exactly as intended.

The total running time of this procedure is $O\left(n \log^2 n\right)$ and it produces at most that many variables and clauses. This leads to the following.

**Theorem 3** *The rectangular bar visibility representation extension problem for a planar $st$-graph with $n$ vertices can be solved in $O\left(n \log^2 n\right)$ time.*

## 5 Hardness Results

In this section we show two hardness results. In the first subsection we show that the bar visibility extension problem for planar graphs is NP-complete. Then, we show that the bar visibility extension problem for directed graphs is NP-complete when restricted to grid representations.

### 5.1 Representations of Undirected Graphs

**Theorem 1** *The Bar Visibility Representation Extension problem is* NP*-complete.*

*Proof* It is clear that the Bar Visibility Representation Extension problem is in NP. To prove completeness, we present a reduction from PlanarMonotone3Sat. Given a formula $\phi$ we construct a graph $G$, a subset $V'$ of vertices of $G$, and a representation $\psi'$ of $V'$ that is extendable to a representation of the whole $G$ if and only if $\phi$ is satisfiable. The vertex $v$ is *fixed* when $v \in V'$, otherwise $v$ is *unrepresented*. The reduction constructs a planar Boolean circuit that simulates the formula $\phi$. The bars assigned to fixed vertices of $G$ create wires and gates of the circuit. Unrepresented vertices of $G$ correspond to Boolean values transmitted over the wires. Our construction uses several Boolean gates: a NOT gate, a XOR gate, a special gate which we call a CXOR gate, and an OR gate.

In the figures illustrating this proof, the red bars denote the fixed vertices of $G$ and the black bars denote the unrepresented vertices. A bar may have its left (right) endpoint marked or not depending on whether the bar extends to the left (right) of the figure or not. The figures also contain some vertical ranges. These ranges are only required for the description of the properties of the gadgets.

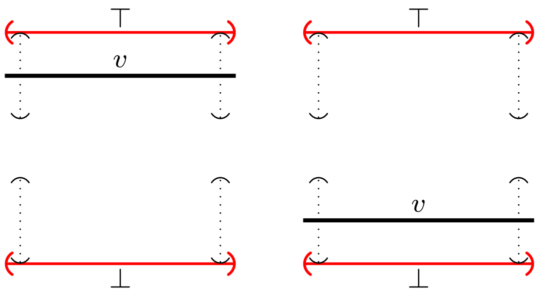

**Fig. 7** Wire transmitting true value (*on the left*) and false value (*on the right*)

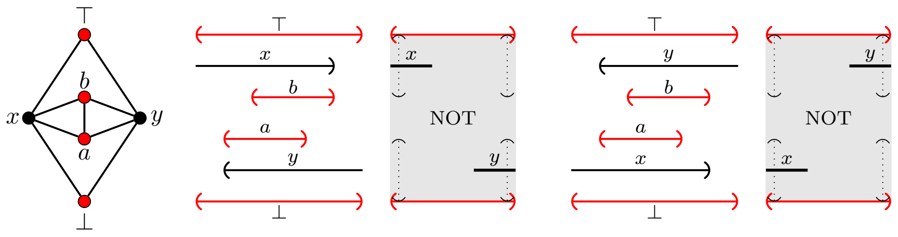

**Fig. 8** The NOT gadget depicted by its two possible representations and their schemes

For readability, the figures contain only schemes of previously defined gadgets. Whenever a scheme appears in a figure, its area is colored gray.

Each wire, see Fig. 7, in the circuit is an empty space between two fixed vertices, $\top$ and $\bot$, whose bars are placed one above the other. One unrepresented vertex $v$, that is adjacent to both $\top$ and $\bot$, corresponds to the value transmitted over the wire. The construction of the gates ensures that in each wire there are exactly two disjoint rectangular regions In the figures, the small horizontal lines with a dotted line between them are used to mark those regions and are not part of the construction. Placement of $v$ anywhere in the top (bottom) region corresponds to a transmission of a true (false) value. We show a collection of gadgets for the gates that use such wires as inputs, and outputs. Similarly to wires, each gate is bounded from the top and bottom by two bars. This way, it is easy to control the visibility between bars from different gates and wires.

**NOT Gadget** Figure 8 presents a NOT gate and its scheme. An unrepresented vertex $x$ ($y$) can transmit the value in the wire that can be placed to the left (right) from the gate. Both bars $a$ and $b$ are adjacent to each other, adjacent to vertices $x$ and $y$, and not adjacent to the bounding bars. As $a$ and $b$ don't have any other neighbors, the only way to obstruct the visibility gap between $a$, $b$ and bounding bars is to use $x$ and $y$. Thus, one of $x$ or $y$ is placed below $a$ and $b$, and the other is placed above $a$ and $b$. This way we obtain a desired functionality of a NOT gate. As the visibility between every

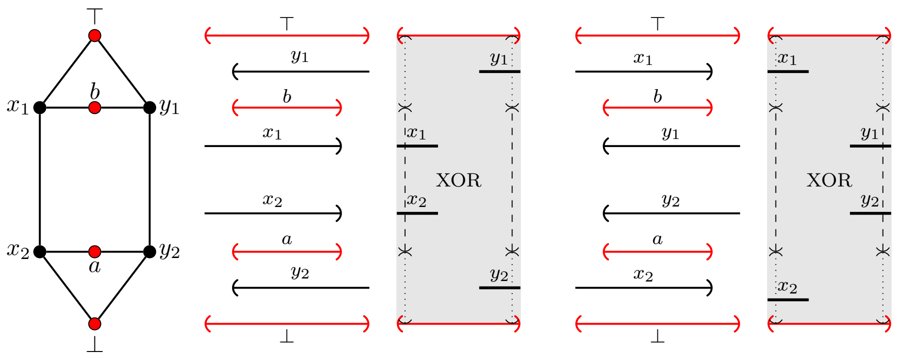

**Fig. 9** The XOR gadget depicted by its two possible representations and their schemes

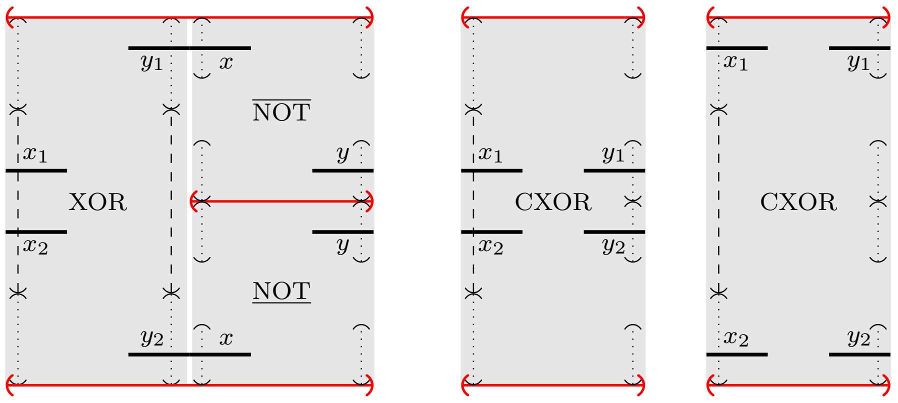

**Fig. 10** The CXOR gadget depicted by the schemes for its two possible representations

two bars does not change across the two representations, the corresponding edges of $G$ are well defined.

**XOR Gadget** Figure 9 presents a XOR gate. It checks that the inputs $x_1$ and $x_2$ have different Boolean values. It also produces outputs $y_1$ and $y_2$. The partial representation is extendable if and only if $x_1 = \neg x_2 = \neg y_1 = y_2$.

To see that, observe that the visibility gap between $b$ and $\top$ needs to be obstructed and $b$ has only two neighbors $x_1$ and $y_1$. Assume that $y_1$ blocks the visibility between $b$ and $\top$. Now $x_1$ needs to block the visibility between $b$ and the other bars. Thus, $x_1$ is placed below $b$. The only other neighbor of $x_1$ is $x_2$ and thus, it needs to go above $a$. The last unrepresented vertex is $y_2$ and it needs to obstruct the visibility gap between $a$ and $\perp$.

The other possibility is that $x_1$ blocks visibility between $b$ and $\top$. The analysis of this case is symmetric to the previous one and gives the second possible valuation of the variables.

Finally, note that the visibility between every two bars does not change across the two representations and the corresponding edges of $G$ are well defined.

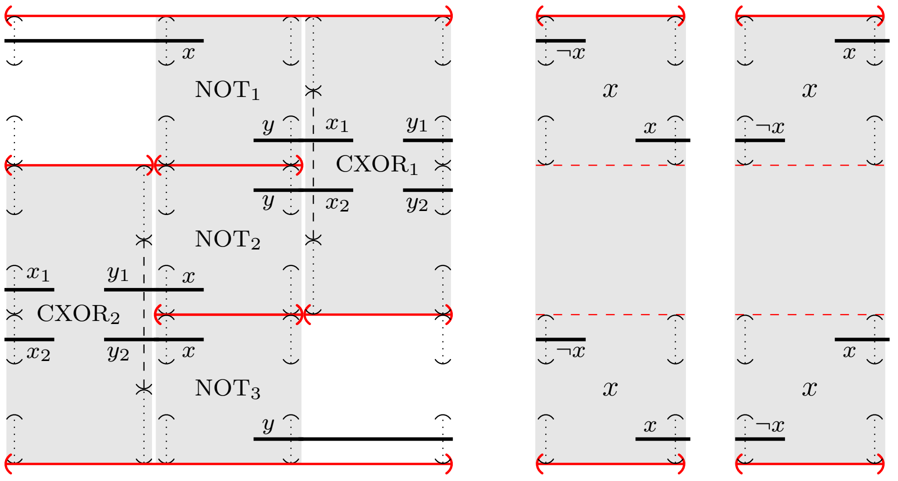

**Fig. 11** *On the left* The gadget for the variable $x$ with two output slots and one of its possible representation for the false value of $x$. *On the right* Schemes of the gadget for the false and the true value of $x$, respectively

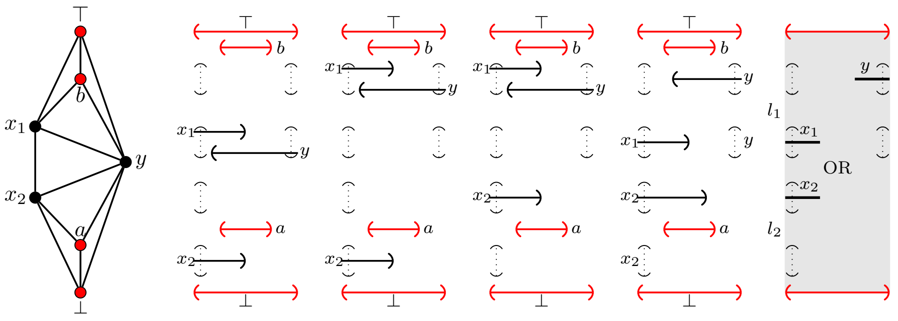

**Fig. 12** The OR gadget

**CXOR Gadget** Figure 10 presents a CXOR circuit. It checks that the inputs $x_1$ and $x_2$ have different Boolean values and produces copies $y_1$ and $y_2$ of the inputs. The CXOR construction combines the XOR gate and two NOT gates in order to obtain a circuit that checks that $x_1 = \neg x_2$, $y_1 = x_1$ and $y_2 = x_2$.

**Variable Gadget** Using NOT gates and CXOR circuits it is easy to construct a variable gadget. Figure 11 presents a gadget that gives two wires with value $x$ to the right side, and two wires with value $\neg x$ to the left side. If we need $k$ copies of a variable, we simply stack $2k - 1$ NOT gates one on another, and add CXOR gates to check the consistency of the outputs produced by every second NOT gate.

All we need to finish the construction of our building blocks is a clause gadget that checks that at least one of three wires connected to it transmits a true value.

**OR Gadget and Clause Gadget** Figure 12 presents an OR gate that has two inputs $x_1$ and $x_2$ and one output $y$.

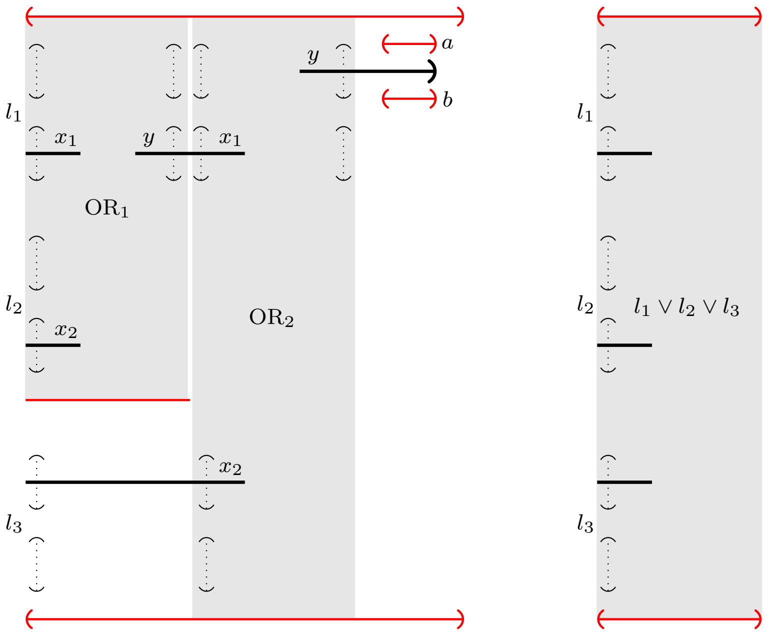

**Fig. 13** The construction for a clause $l_1 \vee l_2 \vee l_3$ and its representation for the false value of $l_1$ and $l_2$ and the true value of $l_3$

The output value can be true only if at least one of the inputs is true. In each of these three scenarios $y$ can be represented in the higher of its regions. See Fig. 12 for these three representations. Now, consider a representation of the gadget where both $x_1$ and $x_2$ are false. Observe that $x_1$ and $b$ are neighbors and we have that $r(x_1) > l(b) = l(a)$. Vertices $x_1$ and $a$ are not adjacent and $x_2$ is represented below $a$. The only other bar that can block the visibility gap between $a$ and $x_1$ is $y$. Therefore, $y$ is placed in the lower of its regions, as it needs to be below $x_1$.

Combining two OR gates and two bars that ensure that the output of the second gate is true, we get a clause gadget presented in Fig. 13.

Given an instance $\phi$ of PlanarMonotone3Sat (together with a rectilinear planar representation of $\phi$), we show how we construct the graph $G$ with a partial representation $\psi'$. We rotate the rectilinear representation by 90 degrees. Now, we replace vertical segments representing variables of $\phi$ with variable gadgets and vertical segments representing clauses of $\phi$ with clause gadgets. Finally, for each occurrence of variable $x$ in clause $C$, we replace the horizontal connection between the segment of $x$ and the segment of $C$ by a wire connecting the appropriate gadgets. The properties of our gadgets assert that $\phi$ is satisfiable if and only if $\psi'$ is extendable to a bar visibility representation of $G$. □

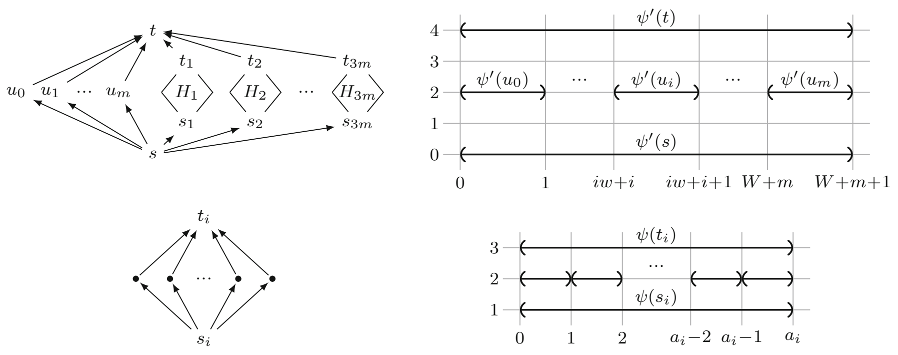

**Fig. 14** The graph $G$, partial representation $\psi'$, graph $H_i$, and the representation $\psi$ of $H_i$ with minimum width

## 5.2 Grid Representations

In this section we consider the following problem:
Grid Bar Visibility Representation Extension for directed graphs:

**Input:** $(G, \psi')$, where $G$ is a directed graph and $\psi'$ is a mapping assigning bars to some subset $V'$ of $V(G)$.

**Question:** Does $G$ admit a grid bar visibility representation $\psi$ with $\psi|V' = \psi'$?
In what follows we show that the above problem is NP-complete.

The proof is generic and can be easily modified to work for other grid representations including undirected, rectangular directed/undirected, and even other models of visibility.

**Theorem 3** *Grid Bar Visibility Representation Extension problem is* NP*-complete.*

*Proof* We use the 3PARTITION problem to show NP-hardness. Consider an instance $w, a_1, \ldots, a_{3m}$ of the 3PARTITION problem. Let $W = \sum_{j=1}^{3m} a_j$, i.e., $W = mw$. From this instance of 3PARTITION, we create a graph $G$ and a partial representation $\psi'$ that assigns bars to a subset $V'$ of $V(G)$. These are constructed as follows and depicted in Fig. 14.

The graph $G$ is constructed as follows. We start with $G'$ which is a $K_{2,m+1}$ with source $s$ and sink $t$ as the two vertices of degree $m + 1$ and the other vertices are labeled $u_0, \ldots, u_m$. Now, for each $i = 1, \ldots, 3m$, create a planar $st$-graph $H_i$ which is a $K_{2,a_i}$ with source $s_i$ and sink $t_i$ as its two vertices of degree $a_i$. We remark that the width of any visibility representation of $H_i$ in the integer grid is at least $a_i$. Finally, the graph $G$ is obtained by attaching each $H_i$ to $G'$ by adding the edges $(s, s_i)$ and $(t_i, t)$.

For the fixed bars, we let $V' = \{s, t, u_1, \ldots, u_{m+1}\}$ and we define a bar $\psi'(v)$ for each element $v \in V'$.

$$y_{\psi'}(v) = \begin{cases} 0 & v = s \\ 2 & v = u_i \\ 4 & v = t \end{cases} \quad l_{\psi'}(v) = \begin{cases} 0 & v = s, t \\ iw + i & v = u_i \end{cases} \quad r_{\psi'}(v) = \begin{cases} W + m + 1 & v = s, t \\ iw + i + 1 & v = u_i \end{cases}$$

Observe that the distance between the right-end of $\psi'(u_i)$ and the left-end of $\psi'(u_{i+1})$ is exactly $w$.

We now claim that there is a solution for input $(G, \psi')$ if and only if the 3Partition instance $\{w, a_1, \dots, a_{3m}\}$ has a solution. First, we consider a collection of triples $T_1, \dots, T_m$ which form a solution of the 3Partition problem. We extend $\psi'$ to a visibility representation of $G$ where, for each triple $T_j = \{a_{j_1}, a_{j_2}, a_{j_3}\}$, we place visibility representations of $H_{j_1}$, $H_{j_2}$, $H_{j_3}$ in sequence left-to-right and between the fixed bars $\psi'(u_{j-1})$ and $\psi'(u_j)$. Notice that this is possible since $a_{j_1} + a_{j_2} + a_{j_3} = w$ and the distance from the right-end of $\psi'(u_{j-1})$ to the left-end of $\psi'(u_j)$ is $w$.

To show that any visibility representation $\psi$ of $G$ which extends $\psi'$ provides a solution to the 3Partition problem we make the following observations. First, due to the the placement of the bars $\psi'(s)$, $\psi'(t)$, $\psi'(u_0)$ and $\psi'(u_m)$, the outer face of the resulting embedding is $s, u_0, t, u_m$. In particular, the bars of every $H_i$ occur strictly within the rectangle enclosed by $\psi'(s)$ and $\psi'(t)$. Thus, each $H_i$ must also be drawn between some pair of bars $\psi'(u_{j-1})$, $\psi'(u_j)$ (for some $j \in \{1, \dots, m-1\}$). Second, due to each $a_i$ being between $\frac{w}{4}$ and $\frac{w}{2}$ and the width of a visibility representation of $H_i$ being at least $a_i$, at most three $H_i$'s 'fit' between the fixed bars $\psi'(u_{j-1})$ and $\psi'(u_j)$. Thus, since there are $3m$ $H_i$'s, every $\psi'(u_{j-1})$, $\psi'(u_j)$ has exactly three $H_i$'s between them. Moreover, if $H_{i_1}$, $H_{i_2}$ and $H_{i_3}$ are placed between $\psi'(u_{j-1})$ and $\psi'(u_j)$, then $a_{i_1} + a_{i_2} + a_{i_3} \leqslant w$. Thus, in $\psi$, the gaps between each pair $\psi'(u_{j-1})$, $\psi'(u_j)$ must contain precisely three $H_i$'s whose sum of corresponding $a_i$'s is $w$, i.e., the gaps correspond to the triples of a solution of the 3Partition problem. □

## 6 Open Problems

The main problem left open is to decide whether there exists a polynomial-time algorithm that checks whether a partial representation of a directed planar graph is extendable to a bar visibility representation of the whole graph. Although we showed an efficient algorithm for the case of planar $st$-graphs, it seems that some additional ideas are needed to resolve this problem in general.

Some further open problems concern extension problems for the weak and strong visibility models.

### 6.1 Weak Visibility

Tamassia and Tollis [47] showed that every planar graph admits a weak visibility representation. Nevertheless, the problem of extending a partial representation of a planar graph to a weak visibility representation is NP-complete [8].

Di Battista and Tamassia [14] showed that a directed planar graph $G$ admits a weak visibility representation if and only if $G$ admits an upward planar drawing. The latter problem is NP-complete [28], so the problem of extending partial weak visibility representations for planar digraphs is also NP-complete. Nevertheless, we do not know whether there is an efficient algorithm if we assume that an upward planar drawing of a planar digraph is given on the input.

### 6.2 Strong Visibility

Due to Andreae [1], the recognition of planar graphs that admit a strong visibility representation is NP-complete. It follows that the problem of testing whether a partial representation of a planar graph is extendable to a strong visibility representation is also NP-complete. Nevertheless, we do not know if there exists an efficient algorithm that tests whether a partial representation of a planar digraph is extendable to a strong representation of the whole graph. It seems that some of our results on the bar visibility model can be adjusted to the strong visibility model.

## References

1. Andreae, T.: Some results on visibility graphs. Discrete Appl. Math. **40**(1), 5–17 (1992)
2. Angelini, P., Di Battista, G., Frati, F., Jelínek, V., Kratochvíl, J., Patrignani, M., Rutter, I.: Testing planarity of partially embedded graphs. ACM Trans. Algorithms **11**(4), 1–42 (2015)
3. Baker, K.A., Fishburn, P.C., Roberts, F.S.: Partial orders of dimension 2. Networks **2**(1), 11–28 (1972)
4. Biedl, T.C.: Small drawings of outerplanar graphs, series-parallel graphs, and other planar graphs. Discrete Comput. Geom. **45**(1), 141–160 (2011)
5. Bläsius, T., Rutter, I.: Simultaneous PQ-ordering with applications to constrained embedding problems. ACM Trans. Algorithms **12**(2), 1–46 (2015)
6. Cardinal, J., Hoffmann, U.: Recognition and complexity of point visibility graphs. Discrete Comput. Geom. **57**(1), 164–178 (2017)
7. Chang, Y.-W., Hutchinson, J.P., Jacobson, M.S., Lehel, J., West, D.B.: The bar visibility number of a graph. SIAM J. Discrete Math. **18**(3), 462–471 (2004)
8. Chaplick, S., Dorbec, P., Kratochvíl, J., Montassier, M., Stacho, J.: Contact representations of planar graphs: Extending a partial representation is hard. In: WG 2014: 40th International Workshop on Graph-Theoretic Concepts in Computer Science, Nouan-le-Fuzelier, France, June 2014. Revised selected papers, pp. 139–151 (2014)
9. Chaplick, S., Fulek, R., Klavík, P.: Extending partial representations of circle graphs. In: GD 2013: 21st International Symposium on Graph Drawing, Bordeaux, France, September 2013. Revised selected papers, pp. 131–142 (2013)
10. Chaplick, S., Guśpiel, G., Gutowski, G., Krawczyk, T., Liotta, G.: The partial visibility representation extension problem. In: GD 2016: 24th International Symposium on Graph Drawing and Network Visualization, Athens, Greece, September 2016. Revised selected papers, pp. 266–279 (2016)
11. Chrobak, M., Payne, T.H.: A linear-time algorithm for drawing a planar graph on a grid. Inf. Process. Lett. **54**(4), 241–246 (1995)

12. Di Battista, G., Eades, P., Tamassia, R., Tollis, I.G.: Graph Drawing: Algorithms for the Visualization of Graphs. Prentice-Hall, Upper Saddle River (1999)
13. Di Battista, G., Frati, F.: A survey on small-area planar graph drawing (2014). arXiv:1410.1006
14. Di Battista, G., Tamassia, R.: Algorithms for plane representations of acyclic digraphs. Theor. Comput. Sci. **61**(2–3), 175–198 (1988)
15. Di Battista, G., Tamassia, R.: On-line planarity testing. SIAM J. Comput. **25**(5), 956–997 (1996)
16. Di Battista, G., Tamassia, R., Tollis, I.G.: Area requirement and symmetry display of planar upward drawings. Discrete Comput. Geom. **7**(1), 381–401 (1992)
17. de Berg, M., Cheong, O., van Kreveld, M., Overmars, M.: Computational Geometry: Algorithms and Applications, 3rd edn. Springer, Berlin (2008)
18. de Berg, M., Khosravi, A.: Optimal binary space partitions for segments in the plane. Int. J. Comput. Geom. Appl. **22**(3), 187–205 (2012)
19. de Fraysseix, H., de Mendez, P.O., Pach, J.: Representation of planar graphs by segments. In: Intuitive Geometry (Szeged, 1991), Volume 63 of Colloquia Mathematica Societatis János Bolyai, pp. 109–117 (1994)
20. de Fraysseix, H., de Mendez, P.O., Rosenstiehl, P.: On triangle contact graphs. Comb. Probab. Comput. **3**, 233–246 (1994)
21. de Fraysseix, H., Pach, J., Pollack, R.: Small sets supporting Fáry embeddings of planar graphs. In: STOC 1988: Proceedings of the 20th Annual ACM Symposium on Theory of Computing, Chicago, IL, USA, pp. 426–433 (1988)
22. de Fraysseix, H., Pach, J., Pollack, R.: How to draw a planar graph on a grid. Combinatorica **10**(1), 41–51 (1990)
23. Driscoll, J.R., Sarnak, N., Sleator, D.D., Tarjan, R.E.: Making data structures persistent. J. Comput. Syst. Sci. **38**(1), 86–124 (1989)
24. Duchet, P., Hamidoune, Y.O., Las Vergnas, M., Meyniel, H.: Representing a planar graph by vertical lines joining different levels. Discrete Math. **46**(3), 319–321 (1983)
25. Fáry, I.: On straight line representation of planar graphs. Acta Univ. Szeged. Sect. Sci. Math. **11**, 229–233 (1948)
26. Garey, M.R., Johnson, D.S.: Computers and Intractability: A Guide to the Theory of NP-Completeness. W.H. Freeman, New York (1979)
27. Garg, A., Tamassia, R.: Upward planarity testing. Order **12**(2), 109–133 (1995)
28. Garg, A., Tamassia, R.: On the computational complexity of upward and rectilinear planarity testing. SIAM J. Comput. **31**(2), 601–625 (2001)
29. Ghosh, S.K., Goswami, P.P.: Unsolved problems in visibility graphs of points, segments, and polygons. ACM Comput. Surv. **46**(2), 1–29 (2013)
30. Gutwenger, C., Mutzel, P.: A linear time implementation of SPQR-trees. In: Proceedings of the GD 2000: 8th International Symposium on Graph Drawing, Colonial Williamsburg, VA, USA, September 2000, pp. 77–90 (2001)
31. Hartman, I.B.-A., Newman, I., Ziv, R.: On grid intersection graphs. Discrete Math. **87**(1), 41–52 (1991)
32. He, X., Wang, J.-J., Zhang, H.: Compact visibility representation of 4-connected plane graphs. Theor. Comput. Sci. **447**, 62–73 (2012)
33. Kant, G., He, X.: Regular edge labeling of 4-connected plane graphs and its applications in graph drawing problems. Theor. Comput. Sci. **172**(1–2), 175–193 (1997)
34. Kant, G., Liotta, G., Tamassia, R., Tollis, I.G.: Area requirement of visibility representations of trees. Inf. Process. Lett. **62**(2), 81–88 (1997)
35. Klavík, P., Kratochvíl, J., Krawczyk, T., Walczak, B.: Extending partial representations of function graphs and permutation graphs. In: Proceedings of the ESA 2012: 20th Annual European Symposium on Algorithms, Ljubljana, Slovenia, September 2012, pp. 671–682 (2012)
36. Klavík, P., Kratochvíl, J., Otachi, Y., Rutter, I., Saitoh, T., Saumell, M., Vyskočil, T.: Extending partial representations of proper and unit interval graphs. Algorithmica **77**(4), 1071–1104 (2017)
37. Klavík, P., Kratochvíl, J., Otachi, Y., Saitoh, T.: Extending partial representations of subclasses of chordal graphs. Theor. Comput. Sci. **576**, 85–101 (2015)
38. Klavík, P., Kratochvíl, J., Otachi, Y., Saitoh, T., Vyskočil, T.: Extending partial representations of interval graphs. Algorithmica 1–23 (2016)
39. Koebe, P.: Kontaktprobleme der konformen Abbildung. Hirzel, Stuttgart (1936)
40. Luccio, F., Mazzone, S., Wong, C.-K.: A note on visibility graphs. Discrete Math. **64**(2–3), 209–219 (1987)

41. Mohar, B.: A polynomial time circle packing algorithm. Discrete Math. **117**(1–3), 257–263 (1993)
42. Myers, E.W.: Efficient applicative data types. In: Proceedings of the POPL 84: 11th ACM SIGACT-SIGPLAN Symposium on Principles of Programming Languages, Salt Lake City, UT, USA, January 1984, pp. 66–75 (1984)
43. Otten, R.H.J.M., van Wijk, J.G.: Graph representations in interactive layout design. In: Proceedings of the IEEE International Symposium on Circuits and Systems, New York, NY, USA, May 1978, pp. 914–918 (1978)
44. Patrignani, M.: On extending a partial straight-line drawing. Int. J. Found. Comput. Sci. **17**(5), 1061–1070 (2006)
45. Schlag, M., Luccio, F., Maestrini, P., Lee, D.-T., Wong, C.-K.: A visibility problem in VLSI layout compaction. Adv. Comput. Res. **2**, 259–282 (1985)
46. Storer, J.A.: On minimal-node-cost planar embeddings. Networks **14**(2), 181–212 (1984)
47. Tamassia, R., Tollis, I.G.: A unified approach to visibility representations of planar graphs. Discrete Comput. Geom. **1**(4), 321–341 (1986)
48. Wang, J.-J., He, X.: Visibility representation of plane graphs with simultaneous bound for both width and height. J. Graph Algorithms Appl. **16**(2), 317–334 (2012)
49. Wismath, S.K.: Characterizing bar line-of-sight graphs. In: Proceedings of the SCG 1985: 1st Annual Symposium on Computational Geometry, Baltimore, MD, USA, June 1985, pp. 147–152 (1985)