Planar Octilinear Drawings with One Bend Per Edge

Michael A. Bekos†, Martin Gronemann‡, Michael Kaufmann†, and Robert Krug§

1Wilhelm-Schickhard-Institut für Informatik, Universität Tübingen, Germany
2Institut für Informatik, Universität zu Köln, Germany

Abstract

In octilinear drawings of planar graphs, every edge is drawn as an alternating sequence of horizontal, vertical and diagonal (45°) line-segments. In this paper, we study octilinear drawings of low edge complexity, i.e., with few bends per edge. A $k$-planar graph is a planar graph in which each vertex has degree less or equal to $k$. In particular, we prove that every 4-planar graph admits a planar octilinear drawing with at most one bend per edge on an integer grid of size $O(n^2) \times O(n)$. For 5-planar graphs, we prove that one bend per edge still suffices in order to construct planar octilinear drawings, but in super-polynomial area. However, for 6-planar graphs we give a class of graphs whose planar octilinear drawings require at least two bends per edge.

1 Motivation

Drawing edges as octilinear paths plays a central role in the design of metro-maps (see e.g., [8, 17, 18]), which dates back to the 1930’s when Henry Beck, an engineering draftsman, designed the first schematic map of London Underground using mostly horizontal, vertical and diagonal segments; see Fig.1. Laying out networks in such a way is called octilinear graph drawing. More precisely, an octilinear drawing of a (planar) graph $G = (V,E)$ of maximum degree eight is a (planar) drawing $\Gamma(G)$ of $G$ in which each vertex occupies a point on the integer grid and each edge is drawn as a sequence of alternating horizontal, vertical and diagonal (45°) line-segments. For an example, see Fig.13 in Section 7.

For drawings of (planar) graphs to be readable, special care is needed to keep the number of bends small. However, the problem of determining whether a given embedded 8-planar graph (that is, a planar graph of maximum degree eight with given combinatorial embedding) admits a bendless octilinear drawing is NP-hard [16]. This negative result motivated us to study octilinear drawings of low edge complexity, that is, with few bends per edge. Surprisingly enough, very few results relevant to this problem were known, even if the octilinear model has been well-studied in the context of metro-map visualization and map schematization (see e.g. [20]). As an immediate byproduct of a result of Keszegh et al. [12], it turns out that every $d$-planar graph, with $3 \leq d \leq 8$, admits a planar octilinear drawing with at most two bends per edge; see Section 2. On the other hand, every 3-planar graph on five or more vertices admits a planar octilinear drawing in which all edges are bendless [5, 11].

In this paper, we bridge the gap between the two aforementioned results. In particular, we prove that every 4-planar graph admits a planar octilinear drawing with at most one bend per edge in cubic

---

*Electronic address: bekos@informatik.uni-tuebingen.de
†Electronic address: gronemann@informatik.uni-koeln.de
‡Electronic address: mk@informatik.uni-tuebingen.de
§Electronic address: krug@informatik.uni-tuebingen.de
area (see Section 4). We further show that every 5-planar graph also admits a planar octilinear drawing with at most one bend per edge, but our construction may require super-polynomial area (see Section 5). Hence, we have made no effort in proving a concrete bound. We complement our results by demonstrating an infinite class of 6-planar graphs whose planar octilinear drawings require at least two bends per edge (see Section 6).

2 Related Work

The research on the (planar) slope number of graphs focuses on minimizing the number of used slopes (see e.g., [9, 12, 13, 14, 15]). Octilinear drawings can be seen as a special case thereof, since only four slopes (horizontal, vertical and the two diagonals) are used. In a related work, Keszegh et al. [12] showed that any \(d\)-planar graph admits a planar drawing with one bend per edge, in which all edge-segments have at most \(2d\) different slopes. So, for \(d = 4\) and for \(d = 5\), we significantly reduce the number of different slopes from 8 and 10, resp., to 4. They also proved that \(d\)-planar graphs, with \(d \geq 3\), admit planar drawings with at most \(\lceil \frac{d}{2} \rceil\) different slopes. It is not difficult to transfer this technique to the octilinear model and show that any \(d\)-planar graph, with \(3 \leq d \leq 8\), admits a planar octilinear drawing with two bends per edge. However, for \(d = 3\), Di Giacomo et al. [5] recently proved that any 3-planar graph with \(n \geq 5\) vertices has a bendless planar drawing with at most 4 different slopes and angular resolution \(\frac{\pi}{4}\) (see also [11]); their approach also yields octilinear drawings.

Octilinear drawings can be considered as an extension of orthogonal drawings, which allow only horizontal and vertical segments (i.e., graphs of maximum degree 4 admit such drawings). Tamassia [19] showed that one can minimize the total number of bends in orthogonal drawings of embedded 4-planar graphs. However, minimizing the number of bends over all embeddings of a 4-planar graph is NP-hard [6]. Note that the core of Tamassia’s approach is a min-cost flow algorithm that first specifies the angles and the bends of the drawing, producing an orthogonal representation, and then based on this representation computes the actual drawing by specifying the exact coordinates for the vertices and the bends of the edges. It is known that Tamassia’s algorithm can be employed to produce a bend-minimum octilinear representation for any given embedded 8-planar graph. However, a bend-minimum octilinear representation may not be realizable by a corresponding planar octilinear drawing [3]. Furthermore, the number of bends on a single edge
might be very high, but can easily be bounded by applying appropriate capacity constraints to the flow-network.

Biedl and Kant [1] showed that any 4-planar graph except the octahedron admits a planar orthogonal drawing with at most two bends per edge on an $O(n^2)$ integer grid. Hence, the oclinlinear drawing model allows us to reduce the number of bends per edge at the cost of an increased area. On the other hand, not all 4-planar graphs admit orthogonal drawings with one bend per edge; however, testing whether a 4-planar graph admits such a drawing can be done in polynomial time [2]. In the context of metro-map visualization, several approaches have been proposed to produce metro-maps using oclinlinear or nearly-octilinear polylines, such as force-driven algorithms [8], hill climbing multi-criteria optimization techniques [18] and mixed-integer linear programs [17]. However, the problem of laying out a metro-map in an octilinear fashion is significantly more difficult than the oclinlinear graph drawing problem, as several metro-lines may connect the same pair of stations and the positions of the vertices have to reflect geographical coordinates of the stations.

3 Preliminaries

In our algorithms, we incrementally construct the drawings similar to the method of Kant [10]. We first employ the canonical order to cope with triconnected graphs. Then, we extend them to biconnected graphs using the SPQR-tree [4] and to simply connected graphs using the BC-tree. In this section we briefly recall them; however we assume basic familiarity.

Definition 1 (Canonical order [10]). For a given triconnected plane graph $G = (V, E)$ let $\Pi = (P_0, \ldots, P_m)$ be a partition of $V$ into paths such that $P_0 = \{v_1, v_2\}$, $P_m = \{v_n\}$ and $v_2 \rightarrow v_1 \rightarrow v_n$ is a path on the outer face of $G$. For $k = 0, \ldots, m$ let $G_k$ be the subgraph induced by $\bigcup_{i=0}^k P_i$ and assume it inherits its embedding from $G$. Partition $\Pi$ is a canonical order of $G$ if for each $k = 1, \ldots, m$ the following hold: (i) $G_k$ is biconnected, (ii) all neighbors of $v_k$ in $G_{k-1}$ are on the outer face, of $G_{k-1}$ (iii) all vertices of $P_k$ have at least one neighbor in $P_j$ for some $j > k$. $P_k$ is called a singleton if $|P_k| = 1$ and a chain otherwise.

Definition 2 (BC-tree). The BC-tree $B$ of a connected graph $G$ has a B-node for each biconnected component of $G$ and a C-node for each cutvertex of $G$. Each B-node is connected with the C-nodes that are part of its biconnected component.

An SPQR-tree [4] provides information about the decomposition of a biconnected graph into its triconnected components. It can be computed in linear time and space [7]. Every triconnected component is associated with a node $\mu$ in the SPQR-tree $\mathcal{T}$. The triconnected component itself is referred to as the skeleton of $\mu$, denoted by $G^{skel}_\mu = (V^{skel}_\mu, E^{skel}_\mu)$. We refer to the degree of a vertex $v \in V^{skel}_\mu$ in $G^{skel}_\mu$ as $deg^{skel}_\mu(v)$. We say that $\mu$ is an R-node, if $G^{skel}_\mu$ is a simple triconnected graph. A bundle of at least three parallel edges classifies $\mu$ as a P-node, while a simple cycle of length at least three classifies $\mu$ as an S-node. By construction R-nodes are the only nodes of the same type that are allowed to be adjacent in $\mathcal{T}$. The leaves of $\mathcal{T}$ are formed by the Q-nodes. Their skeleton consists of two parallel edges; one of them corresponds to an edge of $G$ and is referred to as real edge. The skeleton edges that are not real are referred to as virtual edges. A virtual edge $e$ in $G^{skel}_\mu$ corresponds to a tree node $\mu'$ that is adjacent to $\mu$ in $\mathcal{T}$, more exactly, to another virtual edge $e'$ in $G^{skel}_{\mu'}$. We assume that $\mathcal{T}$ is rooted at a Q-node. Hence, every skeleton (except the one of the root) contains exactly one virtual edge $e = (s, t)$ that has a counterpart in the skeleton of the parent node. We call this edge the reference edge of $\mu$ denoted by $ref(\mu)$.

Its endpoints, $s$ and $t$, are named the poles of $\mu$ denoted by $\mathcal{P}_\mu = \{s, t\}$. Every subtree rooted at a node $\mu$ of $\mathcal{T}$ induces a subgraph of $G$ called the pertinent graph of $\mu$ that we denote by $G^{pert}_\mu = (V^{pert}_\mu, E^{pert}_\mu)$. We abbreviate the degree of a node $v$ in $G^{pert}_\mu$ with $deg^{pert}_\mu(v)$. The pertinent graph is the subgraph of $G$ for which the subtree describes the decomposition.
Now, assume that $G$ be a simple, biconnected $k$-planar graph, whose SPQR-tree $\mathcal{T}$ is given. Additionally, we may assume that $\mathcal{T}$ is rooted at a Q-node that is adjacent to an S- or R-node. Notice that at least one such node exists since the graph does not contain any multi-edges, which would be the case if only a P-node existed. Biconnectivity and maximum degree of $k$ yield basic bounds for the graph degree of a node $v \in V$, i.e., $2 \leq \deg(v) \leq k$. By construction the pertinent graph of a tree node $\mu$ is a (connected) subgraph of $G$; thus $1 \leq \deg_{\mu}^{pert}(v) \leq \deg(v)$. For the degrees in a skeleton graph $G_{\mu}^{skel}$, we obtain bounds based on the type of the corresponding node. Skeletons of Q-nodes are cycles of length two, whereas S-nodes are by definition simple cycles of length at least three; hence, $\deg_{\mu}^{skel}(v) = 2$. For P- and R-nodes the degree can be bounded by $3 \leq \deg_{\mu}^{skel}(v) \leq k$, since the skeleton of the former is at least a bundle of three parallel virtual edges and the latter’s skeleton is triconnected by definition. The upper bound is derived from the relation between skeleton and graph degrees: A virtual edge $e = (s, t)$ hides at least one incident edge of each node (not necessarily an $(s, t)$-edge). This observation can be easily proven by induction on the tree. Hence, $2 \leq \deg_{\mu}^{skel}(v) \leq \deg(v)$.

Next, we use this observation to derive bounds for the pertinent degree by distinguishing two cases depending on whether $v$ is a pole or not. Recall that $G_{\mu}^{pert}$ is a subgraph of $G$ that is obtained by recursively replacing virtual edges by the skeletons of the corresponding children. In the first case when $v$ is an internal node in $G_{\mu}^{pert}$, i.e., $v \notin \mathcal{P}_{\mu}$, $v$ is not incident to the reference edge in $G_{\mu}^{skel}$. Thus, every edge of $G$ hidden by a virtual edge in $G_{\mu}^{skel}$ is in $G_{\mu}^{pert}$. Hence, $\deg_{\mu}^{skel}(v) \leq \deg_{\mu}^{pert}(v) \leq k$.

In the other case, i.e., $v \in \mathcal{P}_{\mu}$, at least one edge that is hidden by the reference edge, is not part of $G_{\mu}^{pert}$, thus, $\deg_{\mu}^{skel}(v) - 1 \leq \deg_{\mu}^{pert}(v) \leq k - 1$. Notice that the lower bounds depend on the skeleton degree which in turn depends on the type of node, unlike the upper bounds that hold for all tree nodes. The next lemma tightens these bounds based on the type of the parent node.

**Lemma 1.** Let $\mu$ be a tree node that is not the root in the SPQR-tree $\mathcal{T}$ of a simple, biconnected, $k$-planar graph $G$ and $\mu'$ its parent in $\mathcal{T}$. For $v \in \mathcal{P}_{\mu}$, it holds that $\deg_{\mu}^{pert}(v) \leq k - 2$, if $\mu'$ is a P- or an R-node and $\deg_{\mu}^{pert}(v) \leq k - 1$ otherwise, i.e. $\mu'$ is an S- or a Q-node.

**Proof.** Since the case where $\mu'$ is an S- or a Q-node follows from the fact that $G$ is $k$-planar and the reference edge hides at least one edge that is not in $G_{\mu}^{pert}$, we restrict ourselves to the more interesting cases where $\mu'$ is either a P- or an R-node. From our previous observations we know that $3 \leq \deg_{\mu'}^{skel}(v) \leq k$. Each of the at least three edges in $G_{\mu'}^{skel}$ hides at least one edge of $G$ that is incident to $v$. However, the total number of edges is at most $k$ due to the degree restriction. Hence, we are left with the problem of $k$ edges of $G$ being hidden by at least three virtual edges, each hiding at least one. As a result the virtual edge that corresponds to $\mu$ cannot contribute more than $k - 2$ edges to its pertinent graph $G_{\mu}^{pert}$. 

**Lemma 2.** In the SPQR-tree $\mathcal{T}$ of a planar biconnected graph $G = (V, E)$ with $\deg(v) \geq 3$ for every $v \in V$, there exists at least one Q-node that is adjacent to a P- or an R-node.

**Proof.** Assume to the contrary that all Q-nodes are adjacent to S-nodes only. We pick such a Q-node and root $\mathcal{T}$ at it. Let $\mu$ be an S-node (possibly the root itself) with poles $\mathcal{P}_{\mu} = \{s, t\}$ such that there is no other S-node in the subtree of $\mu$. By definition of an S-node, $\mu$ has at least two children. If all of them were Q-nodes then there exists a $v \in V_{\mu}^{skel}$ with $s \neq v \neq t$ and $\deg(v) = 2$; a contradiction. Hence, there is at least one child $\mu'$ that is a P- or an R-node. However, since the leaves of $\mathcal{T}$ are Q-nodes and those are not allowed to be children of P- and R-nodes by our assumption, there exists at least one other S-node in the subtree of $\mu'$ and therefore in the subtree of $\mu$ which contradicts our choice of $\mu$. 

4
4 Octilinear Drawings of 4-Planar Graphs

In this section, we focus on planar octilinear drawings of 4-planar graphs. We first consider the case of triconnected 4-planar graphs and then we extend our approach first to biconnected and then to simply connected graphs. Central in our approach is the port assignment; by the port of a vertex we refer to the side of the vertex an edge is connected to. The different ports on a vertex are distinguished by the cardinal directions.

4.1 The Triconnected Case

Let \( G = (V, E) \) be a triconnected 4-planar graph and \( \Pi = \{P_0, \ldots, P_m\} \) be a canonical order of \( G \). We momentarily neglect the edge (distinguished by the cardinal directions. we refer to the side of the vertex an edge is connected to. The different ports on a vertex are simply connected graphs. Central in our approach is the port assignment; by the port of triconnected 4-planar graphs and then we extend our approach first to biconnected and then to

In this section, we focus on planar octilinear drawings of 4-planar graphs. We first consider the case

4 Octilinear Drawings of 4-Planar Graphs

constructed a drawing for \( G \). In case where \( P_{k-1} \) is a singleton is analogous. Having laid out the base of our drawing, we now place in an incremental manner the remaining partitions. Assume that we have already constructed a drawing for \( G_{k-1} \) and we now have to place \( P_k \), for some \( k = 2, \ldots, m - 1 \).

In case where \( P_k = \{v_i, \ldots, v_j\} \) is a chain of \( j - i + 1 \) vertices, we draw them from left to right along a horizontal line one unit above \( G_{k-1} \). Since \( v_i \) and \( v_j \) are the only vertices that are adjacent to vertices in \( G_{k-1} \), both only to one, we place the chain between those two as in Fig.2a. The port used at the endpoints of \( P_k \) in \( G_{k-1} \) depends on the following rule: Let \( v'_l \) (resp.) be the neighbor of \( v_l \) (resp.) in \( G_{k-1} \). If the edge \((v_l, v'_l)\) (resp.) is the last to be attached to vertex \( v_l \) (resp.), i.e., there is no vertex \( v \) in \( P_l \in \Pi, l > k \) such that \((v_l, v) \in E \) (resp.), then we use the northern port of \( v'_l \) (resp.). Otherwise, we choose the north-east port for \((v_l, v'_l)\) or the north-west port for \((v_j, v'_j)\).

In case of a singleton \( P_k = \{v_i\} \), we can apply the previous rule if the singleton is of degree three, as the third neighbor of \( v_i \) should belong to a partition \( \Pi \) for some \( j > k \). However, in case where \( v_i \) is of degree four we may have to deal with an additional third edge \((v_i, v)\) that connects \( v_i \) with \( G_{k-1} \). By the placement so far, we may assume that \( v \) lies between the other two endpoints, thus, we place \( v_i \) such that \( x(v_i) = x(v) \). This enables us to draw \((v_i, v)\) as a vertical line-segment; see Fig.2b.

The above procedure is able to handle all chains and singletons except the last partition \( P_m \), because \( v_n \) may have 4 edges pointing downwards. One of these edges is \((v_n, v_1)\), by definition of \( \Pi \). We exclude \((v_n, v_1)\) and draw \( v_n \) as an ordinary singleton. Then, we shift \( v_1 \) to the left and up as in Fig.2c. This enables us to draw \((v_1, v_n)\) as a horizontal-vertical segment combination. For \((v_1, v_2)\), we move \( v_2 \) one unit to the right and down. We free the west port of \( v_2 \) by redrawing its incident edges as in Fig.2c and attach \((v_1, v_2)\) to it. Edge \((v_1, v_2)\) will be drawn as a diagonal segment with positive slope connected to \( v_1 \) and a horizontal segment connected to \( v_2 \), which requires one bend. Let \((v_2, v_3)\) be the other incomplete edge according to Figure 2c. It will be drawn using a diagonal segment with positive slope connected to \( v_2 \) and a horizontal segment connected to \( v_3 \), again requiring one bend.

So far, we have specified a valid port assignment and the y-coordinates of the vertices. However, we have not fully specified their x-coordinates. Notice that by construction every edge, except the ones drawn as vertical line-segments, contains exactly one horizontal segment. This enables us to stretch the drawing horizontally by employing appropriate cuts. A cut, for us, is a \( y \)-monotone continuous curve that crosses only horizontal segments and divides the current drawing into a left and a right part. It is not difficult to see that we can shift the right part of the drawing that is defined by the cut further to the right while keeping the left part of the drawing on place and the result
remains a valid octilinear drawing.

To compute the x-coordinates, we proceed as follows. We first assign consecutive x-coordinates to the first two partitions and from there on we may have to stretch the drawing in two cases. The first one appears when we introduce a chain, say $P_k$, as it may not fit into the gap defined by its two adjacent vertices in $G_{k-1}$. In this case, we horizontally stretch the drawing between its two adjacent vertices in $G_{k-1}$ to ensure that their horizontal distance is at least $|P_k| + 1$. The other case appears when an edge that contains a diagonal segment is to be drawn. Such an edge requires a horizontal distance between its endpoints that is at least the height it bridges. We also have to prevent it from intersecting any horizontal-vertical combinations in the face below it. We can cope with both cases by horizontally stretching the drawing by a factor that is bounded by the current height of the drawing. Since the height of the resulting drawing is bounded by $|\Pi| = O(n)$, it follows that in the worst case its width is $O(n^2)$. We are now ready to state the main theorem of this subsection.

**Theorem 1.** Given a triconnected 4-planar graph $G$, we can compute in $O(n)$ time an octilinear drawing of $G$ with at most one bend per edge on an $O(n^2) \times O(n)$ integer grid.

**Proof.** In order to keep the time complexity of our algorithm linear, we employ a simple trick. We assume that any two adjacent points of the underlying integer grid are by $n$ units apart in the horizontal direction and by one unit in the vertical direction. This a priori ensures that all edges that contain a diagonal segment will not be involved in crossings and simultaneously does not affect the total area of the drawing, which asymptotically remains cubic. On the other hand, the advantage of this approach is that we can use the shifting method of Kant [10] to cope with the introduction of chains in the drawing, that needs $O(n)$ time in total by keeping relative coordinates that can be efficiently updated and computing the absolute values only at the last step.

Note that our algorithm produces drawings that have a linear number of bends in total (in particular, exactly $2|\Pi| = O(n)$ bends). In the following, we prove that this bound is asymptotically tight.

**Theorem 2.** There exists an infinite class of 4-planar graphs which do not admit bendless octilinear drawings and if they are drawn with at most one bend per edge, then a linear number of bends is required.

**Proof.** Based on the simple fact that in an orthogonal drawing a triangle requires at least one bend, we describe an example that translates this idea to the octilinear model (see Fig.3). While a triangle can easily be drawn bendless with the additional ports available, we will occupy those to enforce the creation of a bend as in the orthogonal model. Furthermore, the example is triconnected. Hence, its embedding is fixed up to the choice of the outer face. Our construction is heavily based on the so called separating triangle, i.e., a three-cycle whose removal disconnects the graph. Each vertex of such a triangle has degree four. Any triangle which is drawn bendless has a $45^\circ$ angle
inside. But since the triangles are nested and have incident edges going inside of the triangles, this is impossible.

4.2 The Biconnected Case

Following standard practice, we employ a rooted SPQR-tree and assume for a tree node that the pertinent graphs of its children are drawn in a pre-specified way. Consider a node $\mu$ in $T$ with poles $\mathcal{P}_\mu = \{s, t\}$. In the drawing of $G^\text{pert}_\mu$, $s$ should be located at the upper-left and $t$ at the lower-right corner of the drawing’s bounding box with a port assignment as in Fig.4a. In general, we assume that the edges incident to $s$ ($t$, resp.) use the western (eastern, resp.) port at their other endpoint, except of the northern (southern, resp.) most edge which may use the north (south, resp.) port instead. In that case we refer to $s$ and $t$ as fixed; see $\overline{\tau}_s$, $\overline{\tau}_t$ in Fig.4a. More specifically, we maintain the following invariants:

IP-1: The width (height) of the drawing of $\mu$ is quadratic (linear) in the size of $G^\text{pert}_\mu$. $s$ is located at the upper-left; $t$ at the lower-right corner of the drawing’s bounding box.

IP-2: If $\deg^\text{pert}_\mu(s) \geq 2$, $s$ is fixed; $t$ is fixed if $\deg^\text{pert}_\mu(t) = 3$ and $\mu$’s parent is not the root.

IP-3: The edges that are incident at $s$ and $t$ in $G^\text{pert}_\mu$ use the south, south-east and east ports at $s$ and the north, north-west and west port at $t$, resp. If $s$ or $t$ is not fixed, incident edges are attached at their other endpoints via the west and east port, respectively. If $s$ or $t$ is fixed, the northern-most edge at $s$ and the southern-most edge at $t$ may use the north (south, resp.) port at its other endpoint.

Notice that the port assignment, i.e. IP-3, guarantees the ability to stretch the drawing horizontally even in the case where both poles are fixed. Furthermore, IP-2 is interchangeable in the following sense: If $\deg^\text{pert}_\mu(s) = 2$ and $\deg^\text{pert}_\mu(t) = 1$, then $s$ is fixed but $t$ is not. But, if we relabel $s$ and $t$ such that $t' = s$ and $s' = t$, then $\deg^\text{pert}_\mu(s') = 1$ and $\deg^\text{pert}_\mu(t') = 2$. By IP-2, we can create a drawing where both $s'$ and $t'$ are not fixed and located in the upper-left and lower-right corner of the drawing’s bounding box. Afterwards, we mirror the resulting layout vertically and horizontally to obtain one where $s$ and $t$ are in their respective corners and not fixed. Notice that in general the property of being fixed is not symmetric, e.g., when $\deg^\text{pert}_\mu(s) = 3$ and $\deg^\text{pert}_\mu(t) = 2$ holds, $s$ remains fixed while $t$ becomes fixed as well. For a non-fixed vertex, we introduce an operation that is referred to as forming or creating a nose; see Fig.4b, where $t$ has been moved downwards at the cost of a bend. As a result, the west port of $t$ is no longer occupied.

P-node case: Let $\mu$ be a P-node. By Lemma 1, for a child $\mu'$ of $\mu$, it holds that $\deg^\text{pert}_\mu(s) \leq 2$ and $\deg^\text{pert}_\mu(t) \leq 2$. So, $t$ can form a nose in $\mu'$, while $s$ might be fixed in the case where $\deg^\text{pert}_\mu'(s) = 2$. Notice that there exists at most one such child due to the degree restriction. We distinguish two cases based on the existence of an $(s, t)$-edge.
For the case where...

**S-node case:** We place the drawings of the children, say $\mu_1, \ldots, \mu_\ell$, of an S-node $\mu$ in a “diagonal manner” such that their corners touch as in Fig.5a. In case of Q-nodes being involved, we draw their edges as horizontal segments (see, e.g., edge $(v_3, v_4)$ in Fig.5a that corresponds to Q-node $\mu_3$). Observe that $s$ and $t$ inherit their port assignment and pertinent degree from $\mu_1$ and $\mu_\ell$, respectively; i.e., $\text{deg}^\text{pert}_\mu(s) = \text{deg}^\text{pert}_{\mu_1}(s)$ and $\text{deg}^\text{pert}_\mu(t) = \text{deg}^\text{pert}_{\mu_\ell}(t)$. So, we may assume that $s$ is fixed in $\mu$, if $s$ is fixed in $\mu_1$. Similarly, $t$ is fixed in $\mu$, if $t$ is fixed in $\mu_\ell$. By IP-2, $t$ is not allowed to be fixed in the case where the parent of $\mu$ is the root of $T$. However, Lemma 2 states that we can choose the root such that $t$ is not fixed in that case, and thus, complies with IP-2. Since we only concatenated the drawings of the children, IP-1 and IP-3 are satisfied.

**R-node case:** For the case where $\mu$ is an R-node with poles $P_\mu = \{s, t\}$, we follow the basic idea of the triconnected algorithm of the previous section and describe the modifications necessary to handle the drawing of the children of $\mu$. To do so, we assume the worst case where no child of $\mu$ is a Q-node. Let $\mu_{\text{unr}}$ denote the child that is represented by the virtual edge $(u, v) \in E^\text{skel}_\mu$. Notice that due to Lemma 1, $\text{deg}^\text{pert}_{\mu_{\text{unr}}}(u) \leq 2$ and $\text{deg}^\text{pert}_{\mu_{\text{unr}}}(v) \leq 2$ holds. Hence, with IP-2 we may assume that at most one out of $u$ and $v$ is fixed in $\mu_{\text{unr}}$. We choose the first partition in the canonical ordering to be $P_0 = \{s, t\}$ and distinguish again between whether the partition to be placed next is a chain or a singleton.

In case of a chain, say $P_k = \{v_i, \ldots, v_j\}$ with two neighbors $v'_i$ and $v'_j$ in $G_{k-1}$, we have to replace two types of edges with the drawings of the corresponding children: the edges $(v_i, v_{i+1}), \ldots, (v_{j-1}, v_j)$ representing the children $\mu_i, \ldots, \mu_{j-1}$ and $(v_j, v_i) ((v_j, v'_i) \text{ resp.})$ representing $\mu'_j (\mu'_i \text{ resp.})$. We place the vertices of $P_k$ on a horizontal line high enough above $G_{k-1}$ such that every drawing may fit in-between it and $G_{k-1}$. Then, we insert the drawings aligned below the horizontal line and choose for $i \leq l < j$, $v_l$ to be the fixed node in $\mu_l$, whereas in $\mu'_j (\mu'_i \text{ resp.})$, we set $v_i (v_j \text{ resp.})$ to be fixed. Hence, for $i \leq l < j$, $v_{l+1}$ may form a nose in $\mu_l$ pointing upwards while $v'_i$ and $v'_j$ form each one downwards as depicted in Fig.5b. For the extra height and width, we stretch the drawing horizontally.
Figure 5: (a) S-node with children $\mu_1, \ldots, \mu_4$; $\mu_3$ is a Q-node representing the edge $(v_3, v_4)$. Optional edges are drawn dotted. (b) Example for a chain $v_i, \ldots, v_j$ with virtual edges representing $\mu_i, \ldots, \mu_{j-1}$ in the R-node case. (c) Singleton $v_i$ with possibly three incident virtual edges representing $\mu'_i, \mu'_e, \mu'_j$. (d) Placing $v_n$ and moving up $s$ which might be fixed in $\mu_{sn}$.

For the case where $P_k = \{v_i\}$ and $i \neq n$ is a singleton, we only outline the difference which is a possible third edge $(v_i, v)$ to $G_{k-1}$ representing say $\mu'_{v}$. While the other two involved children, say $\mu'_i$ and $\mu'_j$, are handled as in the chain-case, $\mu'_v$ requires extra height now and we may place $v_i$ such that $\mu'_v$ fits below $\mu'_j$ as in Fig.5c. Notice that $\deg_{\mu'_v}(v_i) = 1$ holds and therefore by IP-2 both $v_i$ and $v$ are not fixed in $\mu'_{v}$. Hence, forming a nose at $v_i$ and $v$ as in Fig.5c is feasible.

It remains to describe the special case where the last singleton $P_k = \{v_n\}$ is placed. Since $s, t \in P_0$, both have not been fixed yet. We proceed as in the triconnected algorithm and move $s = v_1$ above $v_n$ as depicted in Fig.5d, high enough to accommodate the drawing of the child $\mu_{sn}$ represented by the edge $(s, v_n)$. Since we may require $v_n$ to form a nose in $\mu_{sn}$ as in Fig.5d, we choose $s$ to be fixed in $\mu_{sn}$. However, we are allowed by IP-2 to fix $s$ since $t$ remains unfixed. For the area constraints of IP-1, we argue as follows: Although some diagonal segments may force us to stretch the whole drawing by its height, the height of the drawing has been kept linear in the size of $G_{\mu_{sn}}^{	ext{pert}}$. Since we increase the width by the height a constant number of times per step, the resulting width remains quadratic.

**Root case:** For the root of $T$ we distinguish two cases: In the first case, there exists a vertex $v \in V$ with $\deg(v) \leq 3$. Then, we choose as root a Q-node $\mu$ that represents one of its three incident edges and orient the poles $\{s, t\}$ such that $t = v$. Hence, for the child $\mu'$ of $\mu$ follows $\deg_{\mu'}(t) \leq 2$. In the other case, i.e., for every $v \in V$ we have $\deg(v) = 4$, we choose a Q-node that is not adjacent to an S-node, whose existence is guaranteed by Lemma 2. In both cases, we may form a nose with $t$ pointing downwards and draw the edge as in the triconnected algorithm.

**Theorem 3.** Given a biconnected 4-planar graph $G$, we can compute in $O(n)$ time an octilinear drawing of $G$ with at most one bend per edge on an $O(n^2) \times O(n)$ integer grid.

**Proof.** The SPQR-tree $T$ can be computed in $O(n)$-time and its size is linear to the size of $G$ [7]. The pertinent degrees of the poles at every node can be pre-computed by a bottom-up traversal of $T$. Drawing a P-node requires constant time; S- and R-nodes require time linear to the size of the
skeleton. However, the sum over all skeleton edges is linear, as every virtual edge corresponds to a tree node.

4.3 The Simply Connected Case

After having shown that we can cope with biconnected 4-planar graphs, we turn our attention to the connected case. We start by computing the BC-tree of $G$ and root it at some arbitrary B-node. Every B-node, except the root, contains a designated cut vertex that links it to the parent. A bridge for a biconnected component consists only of a single edge. Similar to the biconnected case, we define an invariant for the drawing of a subtree: The cut vertex that links the subtree to the parent is located in the upper left corner of the drawing’s bounding box.

Any subgraph, say $G_b$, induced by a non-bridge biconnected component can be laid out using the biconnected algorithm. However, to construct a drawing that satisfies our invariant we have to take care of two problems. First, the cut vertex, say $v_b$, that links $G_b$ to the parent, has to be drawn in the upper-left corner of the subtrees drawing. Second, there may be other cut vertices of $G$ in $G_b$ to which we have to attach their corresponding subtrees.

For the first problem we describe how to root the SPQR-tree $T_b$ for $G_b$ so that $v_b$ is located in the upper-left corner. There are at least two Q-nodes having $v_b$ as a pole (as $G_b$ is biconnected) and the degree of $v_b$ in $G_b$ is at most 3. In the biconnected case, we distinguished for the root of the tree between whether there exists $v \in V$ with $\deg(v) \leq 3$ or not. Hence, we may choose for the root of $T_b$ a Q-node having $v_b$ as a pole and orient it such that $v_b = t$, thus, satisfying $\deg(t) \leq 3$. Then, we flip the final drawing of $G_b$ such that $t$ is in the upper left corner (see Fig.6a).

Next, we address the second problem. Let $v_c$ be a cut vertex in $G_b$ that is not the link to the parent. If $v_c$ has degree 3, then it may occur in the pertinent graph of every node. However, in this case we only have to attach a subtree of the BC-tree that is connected via a bridge. This poses no problem, as there are enough free ports available at $v_c$ and we can afford a bend at the bridge. We only consider S- and R-node here since the poles of P-nodes occur in the pertinent graphs of the first two. For R-nodes we assume that the south east port at $v_c$ is free. So, we attach the drawing via the bridge by creating a bend as in Fig.6c. In the diagonal drawing of an S-node, the north-east port is free. So, we can proceed similar; see Fig.6b.

If $v_c$ has degree 2 in $G_b$, it only occurs in the pertinent graph of an S-node; see $v_3$ in Fig.6b. However, we may no longer assume that the bridge is available. As a result, we cannot afford a bend and have to deal with two incident edges instead of one. We modify the drawing by exploiting the two real edges incident to $v_c$ in the S-nodes layout to free the east and south east port; see $v_2$ in Fig.6b. This enables us to attach the subtrees drawing without modifying it. We finish this section by dealing with the most simple case where there are only bridges attached to a cut vertex. The idea is illustrated in Fig.6d and matches our layout specification.

**Theorem 4.** Given a connected 4-planar graph $G$, we can compute in $O(n)$ time an octilinear
5 Octilinear Drawings of 5-Planar Graphs

In this section, we focus on planar octilinear drawings of 5-planar graphs. As in Section 4, we first consider the case of triconnected 5-planar graphs and then we extend our approach first to biconnected and then to the simply connected graphs.

5.1 The Triconnected Case

Let $G = (V, E)$ be a triconnected 5-planar graph and $\Pi = \{P_0, \ldots, P_m\}$ be a canonical order of $G$. We place the first two partitions $P_0$ and $P_1$ of $\Pi$, similar to the case of 4-planar graphs.

Again, we assume that we have already constructed a drawing for $G_{k-1}$ and now we have to place $P_k$, for some $k = 2, \ldots, m - 1$. We further assume that the $x$- and $y$-coordinates are computed simultaneously so that the drawing of $G_{k-1}$ is planar and horizontally stretchable in the following sense: If $e \in E(G_{k-1})$ is an edge incident to the outer face of $G_{k-1}$, then there is always a cut which crosses $e$ and can be utilized to horizontally stretch the drawing of $G_{k-1}$. This is guaranteed by our construction which makes sure that in each step the edges incident to the outer face have a horizontal segment. In other words, one can define a cut through every edge incident to the outer face of $G_{k-1}$ (stretchability-invariant).

If $P_k = \{v_i, \ldots, v_j\}$ is a chain, it is placed exactly as in the case of 4-planar graphs, but with different port assignment. Recall that by $v'_i$ ($v'_j$, resp.) we denote the neighbor of $v_i$ ($v_j$, resp.) in $G_{k-1}$. Among the northern available ports of vertex $v'_i$ ($v'_j$, resp.), edge $\langle v_i, v'_i \rangle$ ($\langle v_j$, $v'_j \rangle$, resp.) uses the eastern-most unoccupied port of $v'_i$ (western-most unoccupied port of $v'_j$, resp.); see Fig.7a. If $P_k$ does not fit into the gap between its two adjacent vertices $v'_i$ and $v'_j$ in $G_{k-1}$, then we horizontally stretch $G_{k-1}$ between $v'_i$ and $v'_j$ to ensure that the horizontal distance between $v'_i$ and $v'_j$ is at least $|P_k| + 1$. This can always be accomplished due to the stretchability-invariant, as both $v'_i$ and $v'_j$ are on the outer face of $G_{k-1}$. Potential crossings introduced by edges of $P_k$ containing diagonal segments can be eliminated by employing similar cuts to the ones presented in the case of 4-planar graphs. So, we may assume that $G_k$ is plane. Also, $G_k$ complies with the stretchability-invariant, as one can define a cut that crosses any of the newly inserted edges of $P_k$ and then follows one of the cuts of $G_{k-1}$ that crosses an edge between $v'_i$ and $v'_j$.

In case of a singleton $P_k = \{v_i\}$ of degree 3 or 4, our approach is very similar to the one of the case of 4-planar graphs. Here, we mostly focus on the case where $v_i$ is of degree five. In this

\[\text{Proof.}\] Decomposing a connected graph into its biconnected components takes linear time. It remains the area property. Inserting a subtree with $n$ vertices and the given dimensions into the drawing of an R- or S-node clearly increases the width of the drawing by at most $O(n^2)$ and the height by at most $O(n)$. Hence, the total drawing area is cubic, as desired.

In case of a singleton $P_k = \{v_i\}$ of degree 5. The eastern-most unoccupied port of $v_i$ belongs to the outer face of $G$ and can be utilized to horizontally stretch the drawing of $G$. So, we may assume that $G_k$ stretch

\[\text{Figure 7: (a) Horizontal placement of a chain } P_k = \{v_i, \ldots, v_j\} \text{ of degree five. (b) Placement of a singleton } P_k = \{v_i\} \text{ of degree five. (c) Final layout (the shape of the dotted edges can be obtained by extending the stubs until they intersect).}\]
Figure 8: A recursive construction of an infinite class of 5-planar graphs requiring super-polynomial drawing area.

case, we have to deal with two additional edges (called nested) that connect \( v_i \) with \( G_{k-1} \), say \((v_1, v)\) and \((v, v')\); see Fig.7b. Such a pair of edges does not always allow vertex \( v_i \) to be placed along the next available horizontal grid line; \( v_i \)'s position is more or less prescribed, as each of \( v \) and \( v' \) may have only one northern port unoccupied. However, a careful case analysis on the type of ports (i.e., north-west, north or north-east) that are unoccupied at \( v \) and \( v' \) in conjunction with the fact that \( G_{k-1} \) is horizontally stretchable shows that we can always find a feasible placement for \( v_i \) (usually far apart from \( G_{k-1} \)). Potential crossings due to the remaining edges incident to \( v_i \) are eliminated by employing similar cuts to the ones presented in the case of 4-planar graphs. So, we may assume that \( G_k \) is planar. Similar to the case of a chain, we prove that \( G_k \) complies with the stretchability-invariant. In this case special attention should be paid to avoid crossings with the nested edges of \( v_i \), as a nested edge may contain no horizontal segment. Note that the case of the last partition \( P_m = \{v_n\} \) is treated in the same way, even if \( v_n \) is potentially incident to three nested edges; see Fig.7c.

To complete the description of our approach it remains to describe how edge \((v_1, v_2)\) is drawn. By construction both \( v_1 \) and \( v_2 \) are along a common horizontal line. So, \((v_1, v_2)\) can be drawn using two diagonal segments that form a bend pointing downwards; see Fig.7c.

**Theorem 5.** Given a triconnected 5-planar graph \( G \), we can compute in \( O(n^2) \) time an octilinear drawing of \( G \) with at most one bend per edge.

**Proof.** Unfortunately, we can no longer use the shifting method of Kant [10], since the \( x \)- and \( y \)-coordinates are not independent. However, the computation of each cut can be done in linear time, which implies that our drawing algorithm needs \( O(n^2) \) time in total.

Recall that when placing a singleton \( P_k = \{v_i\} \) that has four edges to \( G_{k-1} \), the height of \( G_k \) is determined by the horizontal distance of its neighbors along the outer face of \( G_{k-1} \), which is bounded by the actual width of the drawing of \( G_{k-1} \). On the other hand, when placing a chain \( P_k \), the amount of horizontal stretching required in order to avoid potential crossings is delimited by the height of the drawing of \( G_{k-1} \). Unfortunately, this connection implies that for some input triconnected 5-planar graphs our drawing algorithm may result in drawings of super-polynomial area, as the following theorem suggests.

**Theorem 6.** There exist infinitely many triconnected 5-planar graphs for which our drawing algorithm produces drawings of super-polynomial area.

**Proof.** Fig.8 illustrates a recursive construction of an infinite class of 5-planar triconnected graphs with this property. The base of the construction is a “long chain” connecting \( v_1 \) and \( v_2 \) (refer to the bold drawn edges of Fig.8). Each next member, say \( G_{n+O(1)} \), of this class is constructed by adding a constant number of vertices (colored black in Fig.8) to its immediate predecessor member, say \( G_n \).
To compute a layout of an R-node, we employ the triconnected algorithm (with R-node case: drawing of $G$ biconnected 5-planar case. When considering a node $\mu$, polynomial. Since we drop this requirement now we can define a (simpler) new invariant for the 5.2 The Biconnected Case

For the 4-planar case we defined several invariants in order to keep the area of the resulting drawings polynomial. Since we drop this requirement now we can define a (simpler) new invariant for the biconnected 5-planar case. When considering a node $\mu$ in $T$ and its poles $P_{\mu} = \{s, t\}$, then in the drawing of $G_{pert}^\mu$, $s$ and $t$ are horizontally aligned at the bottom of the drawing’s bounding box as in Fig.9a. If an $(s, t)$-edge is present, it can be drawn at the bottom. An $(s, t)$-edge only occurs in the pertinent graph of a P-node (and Q-node). Again, we use the term fixed for a pole-node that is not allowed to form a nose. We maintain the following properties through the recursive construction process: In S- and R- nodes, $s$ and $t$ are not fixed. In P- and Q-nodes, only one of them is fixed, say $s$. But similar to the 4-planar biconnected case, we may swap their roles.

P-node case: Let $\mu$ be a P-node. It is not difficult to see that $\mu$ has at most 4 children; one of them might be a Q-node, i.e., an $(s, t)$-edge, which can be drawn at the bottom as a horizontal segment. Since P-nodes are not adjacent to each other in $T$, the remaining children are S- or R-nodes. By our invariant we may form noses enabling us to stack them as in Fig.9b, as $s$ and $t$ are not fixed in them.

S-node case: Let $\mu$ be an S-node with children $\mu_1, \ldots, \mu_l$. Instead of the diagonal layout used earlier, we now align the drawings horizontally; see Fig.9c. In the S-node case, the poles inherit their pertinent degree from the children and the same holds for the property of being fixed. However, by our new invariant this is forbidden, as it clearly states that $s$ and $t$ are not fixed. It is easy to see that when $\mu_1$ is a P-node, $s$ is fixed by the invariant in $\mu_1$. In this case, we swap the roles of the poles in $\mu_1$ such that $s$ is not fixed. However, the other pole of $\mu_1$, say $v_1$, is fixed now. Since the skeleton of an S-node is a cycle of length at least three, $v_1 \neq t$ holds. As a result, both $s$ and $t$ are not fixed in the resulting drawing.

R-node case: To compute a layout of an R-node, we employ the triconnected algorithm (with $s = v_1$ and $t = v_2$). So, let $\mu$ be an R-node and $\mu_c$ a child of $\mu$ that corresponds to the virtual edge $e = (u, v)$ in $G_{skel}^\mu$. Then, $deg_{pert}^{\mu_c}(u) \leq 3$ and $deg_{pert}^{\mu_c}(v) \leq 3$ holds. When inserting the drawing of $G_{pert}^{\mu_c}$, we require at most three consecutive ports at $u$ and $v$ for the additional edges. As the triconnected algorithm assigns ports in a consecutive manner based on the relative position of the endpoints, we modify the port assignment so that an edge may have more than one port assigned. To do so, we assign each edge $e = (u, v)$ in $G_{skel}^\mu$ a pair $(deg_{pert}^{\mu_c}(u), deg_{pert}^{\mu_c}(v)) \in \{1, 2, 3\}^2$ that reflects the number of ports required by this edge at its endpoints. Then, we extend the triconnected algorithm such that when a port of $u$ is assigned to an edge $e = (u, v)$, $deg_{pert}^{\mu_c}(u) - 1$ additional consecutive ports in clockwise or counterclockwise order are reserved. The direction depends on the different types of edges that we will discuss next.
The simplest type of edges are the ones among consecutive vertices $v_i, v_{i+1}$ of a chain. Recall that $P_0 = \{v_1, v_2\}$ is a special case and the edge $(v_1, v_2)$ is drawn differently. Also, the edges from $P_0$ to $P_1$ are drawn as horizontal segments; see Fig. 7c. For each such edge we reserve the additional ports at $v_i$ in counter-clockwise order and at $v_{i+1}$ in clockwise order; see Fig. 10a. So, we can later plug the drawing of the children into the layout as in Fig. 10b without forming noses. The second type of edges are the ones that connect $P_k = \{v_i, \ldots, v_j\}$ to $v'_i$ and $v'_j$ in $G_{k-1}$. No matter if $P_k$ is a singleton or a chain, we proceed by reserving the ports as in the previous case, i.e., at $v_i$ clockwise, $(v_j$ counter-clockwise, resp.) and at $v'_j$ counter-clockwise $(v'_j$ clockwise); see Fig. 10c. In case where $(v_i, v'_j)$ or $(v'_i, v_j)$ is a virtual edge, we choose the poles such that $v_i$ ($v_j$ resp.) is fixed in $\mu(v_i, v'_j)$ ($\mu(v'_i, v_j)$) resp.). Thus, we can create a nose with $v'_i$ ($v'_j$ resp.). Having exactly the ports required at both endpoints, we insert the drawing by replacing the bend with a nose as in Fig. 10d. The remaining edges from $P_k$ to $G_{k-1}$ in case of a singleton $P_k = \{v_i\}$ can be handled similarly; see Fig. 10. Notice that during the replacement of the edges, the fixed vertex is always the upper one. The only exception are the horizontal drawn edges of a chain. There, it does not matter which one is fixed, as none of the poles has to form a nose.

**Root case:** We root $T$ at an arbitrarily chosen Q-node representing a real edge $(s, t)$. By our invariant we may construct a drawing with $s$ and $t$ at the bottom of the drawing’s bounding box, hence, we draw the edge $(s, t)$ below the bounding box with a ninety degree bend using the south east port at $s$ and south west port at $t$.

**Theorem 7.** Given a biconnected 5-planar graph $G$, we can compute in $O(n^2)$ time an octilinear drawing of $G$ with at most one bend per edge.

**Proof.** We have shown that the ability to rotate and scale suffices to extend the result from 4-planar to 5-planar at the expense of the area. Similar to the 4-planar case, computing $T$ takes linear time. Hence, the overall runtime is governed by the triconnected algorithm.

### 5.3 The Simply Connected Case

In the following, we only outline the differences in comparison with the corresponding 4-planar case. As an invariant, the drawing of every subtree should conform to the layout depicted in Fig. 11a. For a single biconnected component $b$, let $v_c$ refer to the cut vertex linking it to the parent. As root for the SPQR-tree $T_b$ of $G_b$, we again choose a Q-node $\mu_r$, whose real edge is incident to $v_c$; see Fig. 11b. Hence, the layout generated by the biconnected approach matches this scheme.

It remains to show that we can attach the children. Since we are able to scale and rotate, we keep things simple and look for suitable spots to attach them. Notice that in the drawings of S-nodes and
chains in R-nodes all southern ports are free. Hence, we may rotate the drawings of the subtrees and attach the at most three (two for a chain) edges to \( v_c \) there (refer to Fig.11c for an example of a chain). The only exception are the singletons. Assume that \( v_i \) is an ordinary singleton that has one nested edge attached. Hence, it has degree four, leaving us with a single bridge to attach the component; Fig.11d. However, this does not hold in case \( v_i = v_n \). Consider the case where \( v_n \) has a nested edge and we have to attach a subtree that requires two ports. As a result \( v_n \) has degree 3 in \( G_b \) and, thus, all northern ports are free.

**Theorem 8.** Given a connected 5-planar graph \( G \), we can compute in \( O(n^2) \) time an octilinear drawing of \( G \) with at most one bend per edge.

**Proof.** We described how to attach any subtree to cut vertices inside a biconnected component. Furthermore, the component itself complies with the layout scheme. In addition, this scheme enables us to compose such drawings at a cut vertex using rotations. \( \Box \)

6 A Note on Octilinear Drawings of 6-Planar Graphs

In this section, we show that it is not always possible to construct a planar octilinear drawing of a given 6-planar graph with at most one bend per edge. In particular, we present an infinite class of 6-planar graphs, which do not admit planar octilinear drawings with at most one bend per edge.

**Theorem 9.** There exists an infinite class of 6-planar graphs which do not admit planar octilinear drawings with at most one bend per edge.

**Proof.** Our proof is heavily based on the following simple observation: If the outer face \( F(\Gamma(G)) \) of a given planar octilinear drawing \( \Gamma(G) \) consists of exactly three vertices, say \( v, v' \) and \( v'' \), that have the so-called outerdegree-property, i.e., \( \deg(v) = \deg(v') = 6 \) and \( 5 \leq \deg(v'') \leq 6 \), then it is not feasible to draw all edges delimiting \( F(\Gamma(G)) \) with at most one bend per edge; one of them has to be drawn with (at least) two bends in \( \Gamma(G) \). Next, we construct a specific maximal 6-planar graph, in which each face has at most one vertex of degree 5 and at least two vertices of degree 6; see Fig.12a. This specific graph does not admit a planar octilinear drawing with at most one bend, as its outerface is always bounded by three vertices that have the outerdegree-property.

To obtain an infinite class of 6-planar graphs with this property, we give the following recursive construction. Let \( G_1 \) and \( G_2 \) be two copies of the graph of Fig.12a. Let also \( f_i \) be a bounded face of \( G_i \), \( i = 1, 2 \). We proceed to subdivide each edge of \( f_i \) by introducing a new vertex on it. We further assume that the new vertices of \( f_i \) are pairwise adjacent (see the top part of Fig.12b). Hence, they form a triangular face, say \( f'_i \), in the augmented graph, say \( G'^{aug}_i \), constructed in this manner. Up to now, each of the newly introduced vertices is of degree four. Now, assume that \( G'^{aug}_1 \) is drawn on the plane so that \( f'_1 \) is a bounded face in \( \Gamma(G'^{aug}_1) \), and \( G'^{aug}_2 \) is drawn such that \( f'_2 \) is the unbounded face in \( \Gamma(G'^{aug}_2) \). By choosing \( f'_2 \) as the outer face in \( \Gamma(G'^{aug}_2) \) we make sure that we can connect the three degree four vertices of \( f'_2 \) to the three degree four vertices of \( f'_1 \) in
Figure 12: (a) A maximal 6-planar graph in which each face has at most one vertex of degree 5 (black-colored vertices) and at least two vertices of degree 6 (gray-colored vertices). From Euler’s formula for maximal planar graphs, it follows that any graph with this property must have at least 12 vertices of degree 5. Hence, this is the smallest graph with this property. (b) Illustration of the recursive construction.

the following way: We appropriately scale down $\Gamma(G'_{2}^{aug})$ and proceed to draw it in the interior of $f'_1$ without introducing any crossings (see the small gray-colored triangle of the bottom drawing of Fig.12b). If we connect the vertices of $f'_1$ and $f'_2$ in an “octahedron-like manner”, then all vertices of $f'_1$ and $f'_2$ are of degree 6 and the resulting graph, say $G_1^{aug} \oplus G_2^{aug}$, is maximal 6-planar and has the outerdegree-property.

7 A Sample Octilinear Drawing with at most 1 bend per edge

Figure 13: Example layout of a biconnected 4-planar graph. Vertices are labeled by their indices. The corresponding SPQR-tree $T$ has been rooted at a Q-node representing the edge $(v_{32}, v_{2})$ with the only child being an S-node whose skeleton is the simple cycle $v_{32}, v_{21}, v_{2}$. It has two R-nodes as children, a smaller in the upper left (with poles $\{v_{32}, v_{21}\}$) and a larger one (with poles $\{v_{21}, v_{2}\}$) occupying most of the drawing area. The latter one contains two smaller S-nodes (with poles $\{v_{10}, v_{12}\}$ and $\{v_{4}, v_{26}\}$) and a P-node (with poles $\{v_{26}, v_{25}\}$) that has two children. One of them being an $(s,t)$-edge, the other one an S-node.
8 Conclusions

Motivated by the fundamental role of planar octilinear drawings in map schematization, we presented algorithms for their construction with at most one bend per edge for 4- and 5-planar graphs. We also improved the known bounds on the required number of slopes for 4- and 5-planar drawings from 8 and 10, resp. ([12]) to 4. Our work raises several open problems:

- Is it possible to construct planar octilinear drawings of 4-planar (5-planar, resp.) graphs with at most one bend per edge in $o(n^3)$ (polynomial, resp.) area?
- Does any triangle-free 6-planar graph admit a planar octilinear drawing with at most one bend per edge?
- What is the complexity to determine whether a 6-planar graph admits a planar octilinear drawing with at most one bend per edge?
- What is the number of necessary slopes for bendless drawings of 4-planar graphs?

References


