Optimal Orthogonal Graph Drawing with Convex Bend Costs*

Thomas Bläsius, Ignaz Rutter, and Dorothea Wagner

Faculty of Informatics, Karlsruhe Institute of Technology (KIT), Germany
firstname.lastname@kit.edu

Abstract. Traditionally, the quality of orthogonal planar drawings is quantified by the total number of bends, or the maximum number of bends per edge. However, this neglects that in typical applications, edges have varying importance. We consider the problem OptimalFlexDraw that is defined as follows. Given a planar graph G on n vertices with maximum degree 4 (*4-planar graph*) and for each edge e a cost function $\text{cost}_e \colon \mathbb{N}_0 \longrightarrow \mathbb{R}$ defining costs depending on the number of bends e has, compute an orthogonal drawing of G of minimum cost.

In this generality OptimalFlexDraw is NP-hard. We show that it can be solved efficiently if 1) the cost function of each edge is convex and 2) the first bend on each edge does not cause any cost. Our algorithm takes time $O(n \cdot T_{\text{flow}}(n))$ and $O(n^2 \cdot T_{\text{flow}}(n))$ for biconnected and connected graphs, respectively, where $T_{\text{flow}}(n)$ denotes the time to compute a minimum-cost flow in a planar network with multiple sources and sinks. Our result is the first polynomial-time bend-optimization algorithm for general 4-planar graphs optimizing over all embeddings. Previous work considers restricted graph classes and unit costs.

1 Introduction

Orthogonal graph drawing is one of the most important techniques for the human-readable visualization of complex data. Since edges are required to be straight orthogonal lines—which automatically yields good angular resolution and short links—the human eye may easily adapt to the flow of an edge. The readability of orthogonal drawings can be further enhanced in the absence of crossings, i.e., if the underlying data exhibits planar structure. In order to be able to visualize all 4-planar graphs, we allow edges to have bends. Since bends decrease readability, we seek to minimize the number of bends.

We consider the problem OptimalFlexDraw whose input consists of a planar graph G with maximum degree 4 and for each edge e a cost function $\text{cost}_e \colon \mathbb{N}_0 \longrightarrow \mathbb{R}$ defining costs depending on the number of bends on e. We seek an orthogonal drawing of G with minimum cost. Garg and Tamassia [7] show that it is NP-hard to decide whether a 4-planar graph admits an orthogonal drawing with zero bends, directly implying that OptimalFlexDraw is NP-hard in

* Partly done within GRADR – EUROGIGA project no. 10-EuroGIGA-OP-003.

F.V. Fomin et al. (Eds.): ICALP 2013, Part I, LNCS 7965, pp. 184–195, 2013.

general. For special cases, namely planar graphs with maximum degree 3 and series-parallel graphs, Di Battista et al. [4] give an algorithm minimizing the total number of bends, optimizing over all planar embeddings. They introduce the concept of spirality that is similar to the concept of rotation we use. Bläsius et al. [2] consider the decision problem FlexDraw, where each edge has a *flexibility* specifying its allowed number of bends. They give a polynomial-time decision algorithm for the case that all flexibilities are positive. OptimalFlexDraw can be seen as the optimization version of FlexDraw since it allows to find a drawing that minimizes the number of bends exceeding the flexibilities.

As minimizing the total number of bends is NP-hard, many results initially fix the planar embedding. Tamassia [9] describes a flow network for minimizing the number of bends for a fixed planar embedding. The technique can be easily adapted to solve OptimalFlexDraw if the planar embedding is fixed. Biedl and Kant [1] show that every planar graph admits a drawing with at most two bends per edge except for the octahedron. Even though fixing an embedding allows to efficiently minimize the total number of bends, it neglects that this choice may have a huge impact on the number of bends in the resulting drawing.

Contribution and Outline. Our main result is the first polynomial-time bend-optimization algorithm for general 4-planar graphs optimizing over all embeddings. Previous work considers restricted graph classes and unit costs. We solve OptimalFlexDraw if 1) all cost functions are convex and 2) the first bend is for free. Note that convexity is quite natural, and without condition 2) OptimalFlexDraw is NP-hard. An interesting special case is the minimization of the total number of bends over all planar embeddings, where one bend is for free. Moreover, as every 4-planar graph has a drawing with at most two bends per edge [1], we can minimize the number of 2-bend edges in such a drawing.

To solve OptimalFlexDraw for biconnected graphs, we extend the notion "number of bends" to split components and use dynamic programming to compute their cost functions bottom-up in the SPQR-tree. In each step we use a flow network similar to the one described by Tamassia [9]. The major problem is that the cost functions for split components may be non-convex [3]. To overcome this problem, we show the existence of an optimal solution with at most three bends per edge except for a single edge per block with up to four bends. Due to an extension to split components, it suffices to consider their cost functions on the interval $[0, 3]$, and we show that, on this interval, they are convex.

We show in Section 3 that for biconnected graphs, the number of bends per edge can always be reduced to three and generalize this result to split components in Section 4. In Section 5 we show that the cost functions for split components are convex on the interval $[0, 3]$. This yields an algorithm for computing optimal drawings of biconnected graphs, which extends to connected graphs. Omitted proofs are in the appendix and in the full version of this paper [3].

2 Preliminaries

An instance of OptimalFlexDraw is a 4-planar graph G together with a cost function $\mathrm{cost}_e : \mathbb{N}_0 \longrightarrow \mathbb{R} \cup \{\infty\}$ for each edge e assigning a cost to e depending on the number of its bends. OptimalFlexDraw asks for an optimal orthogonal drawing, i.e., a drawing with minimum cost summed over all edges.

For a cost function $\text{cost}_e(\cdot)$ let $\Delta\,\text{cost}_e(\rho) = \text{cost}_e(\rho+1) - \text{cost}_e(\rho)$ be its *difference function*. A cost function is *monotone* if its difference function is greater or equal to 0. It is *convex*, if its difference function is monotone. The *base cost* of the edge e with monotone cost function is $b_e = \text{cost}_e(0)$. According to the decision problem FlexDraw, G is said to have *positive flexibility* if $\text{cost}_e(0) = \text{cost}_e(1)$ holds for every edge e. An instance G of OptimalFlexDraw is *positive-convex* if it has positive flexibility and each cost function is convex.

2.1 Connectivity and the SPQR-Tree

A graph is *connected* if there exists a path between any pair of vertices. A *separating k-set* is a set of k vertices whose removal disconnects the graph. Separating 1-sets and 2-sets are *cutvertices* and *separation pairs*, respectively. A connected graph is *biconnected* (*triconnected*) if it does not have a cutvertex (separation pair). The *cut components* with respect to a separating k-set S are the maximal subgraphs that are not disconnected by removing S.

The *SPQR-tree* $\mathcal{T}$ introduced by Di Battista and Tamassia [5,6] is a succinct representation of all planar embeddings of a biconnected planar graph G. It describes a decomposition of G along its *split pairs*, which are separation pairs or single edges, into triconnected components. It can be computed in linear time [8] and has linear size. Every node μ of $\mathcal{T}$ is associated with a multigraph $\text{skel}(\mu)$, called *skeleton*, on a subset of the vertices of G. Each inner node in $\mathcal{T}$ is an S-, P- or R-node, having a cycle, a bunch of parallel edges and a triconnected graph as skeleton, respectively. The edges in these skeletons are called *virtual edges*. The leaves of $\mathcal{T}$ are Q-nodes, their skeletons consist of an edge of G plus a parallel virtual edge. When two nodes μ_1 and μ_2 are adjacent in $\mathcal{T}$ this edge identifies a virtual edge in $\text{skel}(\mu_1)$ with a virtual edge in $\text{skel}(\mu_2)$, and each virtual edge in each node is associated with exactly one such neighbor.

Rooting the SPQR-tree in some node τ determines for each node $\mu \neq \tau$ a unique *parent edge* in $\text{skel}(\mu)$ that is associated with μ's parent. The *pertinent graph* $\text{pert}(\mu)$ of a node μ is recursively defined as follows. For a Q-node $\text{pert}(\mu)$ is the edge in G it corresponds to. For an inner node $\text{pert}(\mu)$ is the graph obtained from $\text{skel}(\mu)$ by deleting the parent edge and replacing each virtual edge by the pertinent graph of the corresponding child. The *expansion graph* of a virtual edge ε is the pertinent graph of the child of μ corresponding to ε when $\mathcal{T}$ is rooted at μ. The SPQR-tree represents all embeddings of G on a sphere, i.e., embeddings without a specific outer face, by allowing independent choices for the embeddings of all skeletons. For R-nodes the embedding is fixed up to a flip, for P-nodes one can choose an arbitrary order for the parallel edges, and for S- and Q-nodes there is no embedding choice.

Usually the SPQR-tree is assumed to be unrooted, as described above, or rooted at a Q-node, representing embeddings with the corresponding edge on the outer face. We consider the SPQR-tree to be rooted at an arbitrary node τ. This also restricts the choice of the outer face and the embedding choices are of the following kind. For every node $\mu \neq \tau$ one can choose an embedding for skel(μ) with the parent edge on the outer face. For τ itself, the choice for the embedding of skel(τ) includes the choice of an outer face.

2.2 Orthogonal Representation

Two orthogonal drawings of a 4-planar graph G are *equivalent*, if they have the same planar embedding, and the same shape, i.e., the sequence of right and left turns is the same when traversing the faces of G. To make this precise, we define *orthogonal representations*, originally introduced by Tamassia [9], as equivalence classes of this relation. To ease the notation we only consider biconnected graphs.

Let Γ be an orthogonal drawing of a biconnected 4-planar graph G and let $\mathcal{E}$ be the planar embedding induced by it. We define the *rotation* of an edge e in an incident face f to be the number of bends to the right minus the number of bends to the left when traversing f in clockwise order (counter-clockwise if f is the outer face) and denote the resulting value by $\mathrm{rot}(e_f)$. Similarly, we define the rotation of a vertex v in an incident face f, denoted by $\mathrm{rot}(v_f)$, to be 1, -1 and 0 if there is a turn to the right, a turn to the left and no turn, respectively. The orthogonal representation $\mathcal{R}$ belonging to Γ consists of the planar embedding $\mathcal{E}$ of G and all rotation values of edges and vertices, respectively. It is easy to see that every orthogonal representation has the following properties.

(I) For every edge e with incident faces f_1, f_2 we have $\mathrm{rot}(e_{f_1}) = -\mathrm{rot}(e_{f_2})$.
(II) The sum over all rotations in a face is 4 (-4 for the outer face).
(III) The sum of rotations around a vertex v is $2 \cdot (\deg(v) - 2)$.

Tamassia showed that the converse is also true [9], i.e., if $\mathcal{R}$ satisfies the above properties, then it is the orthogonal representation of a class of drawings. In what follows we always neglect the exact geometry and work with orthogonal representations instead of drawings. In some cases we write $\mathrm{rot}_{\mathcal{R}}(\cdot)$ instead of $\mathrm{rot}(\cdot)$ to make clear which orthogonal representation we refer to. Moreover, the face in the subscript is omitted if it is clear from the context.

Let G be a 4-planar graph with orthogonal representation $\mathcal{R}$ and two vertices s and t incident to a common face f. We define $\pi_f(s,t)$ to be the path from s to t on the boundary of f, when traversing f in clockwise direction (counter-clockwise if f is the outer face). Let $s = v_1, \dots, v_k = t$ be the vertices on the path $\pi_f(s,t)$. The rotation of $\pi(s,t)$ is defined as

$$\mathrm{rot}(\pi(s,t)) = \sum_{i=1}^{k-1} \mathrm{rot}(\{v_i, v_{i+1}\}) + \sum_{i=2}^{k-1} \mathrm{rot}(v_i)\,.$$

Let G be a biconnected positive-convex instance of OptimalFlexDraw with optimal orthogonal representation $\mathcal{R}$ and let H be a split component with respect to $\{s,t\}$ such that the orthogonal representation $\mathcal{S}$ induced by H has s and t on its outer face. Then $\mathcal{S}$ is *tight* with respect to s and t if the rotations of s and t in internal faces are 1, i.e., they have 90°-angles in internal faces. The orthogonal representation of G is *tight* if every split component having its corresponding split pair on its outer face is tight. We can assume without loss of generality that all orthogonal representations are tight [2, Lemma 2].

2.3 Flow Network

A *flow network* is a tuple $N = (V, A, \mathrm{COST}, \mathrm{dem})$ where (V, A) is a directed (multi-)graph, COST is a set containing a *convex cost function* $\mathrm{cost}_a\colon \mathbb{N}_0 \longrightarrow \mathbb{R} \cup \{\infty\}$ for each arc $a \in A$ and $\mathrm{dem}\colon V \longrightarrow \mathbb{Z}$ is the *demand* of the vertices. A *flow* is a function $\phi\colon A \longrightarrow \mathbb{N}_0$ assigning a certain amount of flow to each arc. It is *feasible*, if the difference of incoming and outgoing flow at each vertex equals its demand. The *cost* of ϕ is $\mathrm{cost}(\phi) = \sum_{a \in A} \mathrm{cost}_a(\phi(a))$. An arc a has *capacity* c if $\mathrm{cost}_a(\rho) = 0$ for $\rho \in [0, c]$ and $\mathrm{cost}_a(\rho) = \infty$ otherwise.

The *parameterized flow network* with respect to two nodes $u, v \in V$ is defined the same as N but with a *parameterized demand* of $\mathrm{dem}(u) - \rho$ for u and $\mathrm{dem}(v) + \rho$ for v where ρ is a parameter. The *cost function* $\mathrm{cost}_N(\rho)$ of N is defined to be $\mathrm{cost}(\phi)$ of an optimal flow ϕ in N with respect to the demands determined by ρ. Increasing ρ by 1 can be seen as pushing one unit of flow from u to v.

Theorem 1. *The cost function of a parameterized flow network is convex on the interval* $[\rho_0, \infty]$, *where* $\rho_0 = \mathrm{argmin}_{\rho \in \mathbb{Z}}\{\mathrm{cost}_N(\rho)\}$.

3 Valid Drawings with Fixed Planar Embedding

In this section we consider the problem FlexDraw for biconnected planar graphs with fixed embedding. Given an arbitrary *valid orthogonal representation*, i.e., an orthogonal representation that respects the flexibilities, we show the existence of a valid orthogonal representation with the same angles around vertices, the same planar embedding, and at most three bends per edge except for possibly a single edge on the outer face with up to five bends.

Let G be a 4-planar graph with positive flexibility and valid orthogonal representation $\mathcal{R}$. If the number of bends of an edge e equals its flexibility, we orient e such that its bends are right bends (we always assume that edges are bent in only one direction). Otherwise, e remains undirected. A path $\pi = (v_1, \dots, v_k)$ in G is *directed*, if the edge $\{v_i, v_{i+1}\}$ (for $i \in \{1, \dots, k-1\}$) is either undirected or directed from v_i to v_{i+1}. It is *strictly directed*, if it is directed and does not contain undirected edges. These definitions extend to *(strictly) directed cycles*. The terms $\mathrm{left}(C)$ and $\mathrm{right}(C)$ denote the set of edges and vertices lying to the left and right of a (strictly) directed cycle C. A cut $(U, V \setminus U)$ is *directed* from U to $V \setminus U$, if every edge crossing the cut is undirected or directed from U to $V \setminus U$. It is *strictly directed* if it additionally does not contain undirected edges.

Lemma 1. *Let G be a graph with positive flexibility and vertices s and t such that $G + st$ is biconnected and 4-planar. Let further $\mathcal{R}$ be a valid orthogonal representation with s and t incident to a face f such that $\pi_f(t, s)$ is strictly directed from t to s. The following holds.*
(1) $\mathrm{rot}_{\mathcal{R}}(\pi_f(s, t)) \le -3$ *if f is the outer face and G is not a single path*
(2) $\mathrm{rot}_{\mathcal{R}}(\pi_f(s, t)) \le -1$ *if f is the outer face*
(3) $\mathrm{rot}_{\mathcal{R}}(\pi_f(s, t)) \le 5$

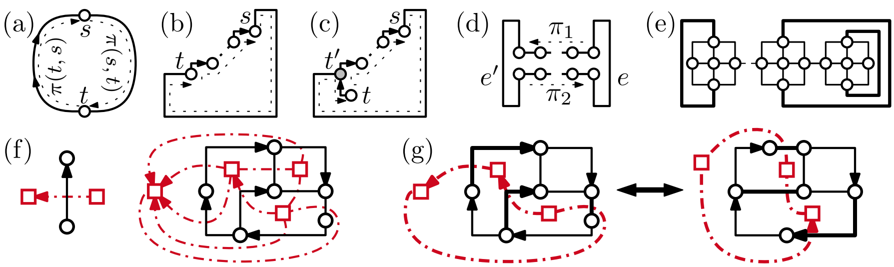

Fig. 1. (a–c) Illustration of Lemma 1. (d) Two edges on the outer face with four bends. (e) Example with $O(n)$ edges requiring four bends. (f) The flex graph of an orthogonal drawing. (g) Bending along a cycle in the flex graph. The resulting flex graph contains the same cycle directed in the opposite direction, so this operation can be reversed.

Proof (Sketch). We show case (3), where f is an internal face; see Fig. 1(a). The other cases work similarly. Since $\pi_f(t,s)$ is strictly directed, every edge on this path has at least one right bend (when traversing from t to s), yielding a rotation of at least 1. Moreover, every internal vertex in $\pi_f(t,s)$ may have a left bend, yielding a rotation of at most -1. As the number of internal vertices is one less than the number of edges in a path, $\mathrm{rot}(\pi_f(t,s)) \geq 1$ holds. We first assume that neither s nor t have degree 1; see Fig. 1(b). As the rotation around f is 4, we have $\mathrm{rot}(\pi_f(s,t)) = 4 - \mathrm{rot}(s_f) - \mathrm{rot}(t_f) - \mathrm{rot}(\pi_f(t,s))$. Moreover, we have $\mathrm{rot}(s_f), \mathrm{rot}(t_f) \geq -1$ (since $\deg(s), \deg(t) > 1$) and $\mathrm{rot}(\pi_f(t,s)) \geq 1$ (as seen above), yielding $\mathrm{rot}(\pi_f(s,t)) \leq 5$. If t (or s) has degree 1, $\mathrm{rot}(\pi_f(t,s)) \geq 2$ holds since an internal vertex t' (or s') of $\pi_f(t,s)$ has degree 3 and thus cannot have rotation -1, canceling out the rotation of -2 at t (or s); see Fig. 1(c). □

The *flex graph* $G^\times_\mathcal{R}$ of G with respect to a valid orthogonal representation $\mathcal{R}$ is the dual graph of G such that the dual edge $e^\star$ is directed from the face right of e to the face left of e (or undirected if e is undirected); see Fig. 1(f). Assume C is a simple directed cycle in the flex graph. Then *bending* along this cycle yields a new valid orthogonal representation $\mathcal{R}'$ defined as follows; see Fig. 1(g). For an edge $e^\star = (f_1, f_2)$ in C dual to e we decrease $\mathrm{rot}(e_{f_1})$ and increase $\mathrm{rot}(e_{f_2})$ by 1. It is easy to see that $\mathcal{R}'$ is an orthogonal representation. Moreover, no edge has more bends than allowed by its flexibility, as C is directed. The following lemma states that a high rotation along a path $\pi_f(s,t)$ for two vertices s and t sharing the face f can be reduced by 1 using a directed cycle in the flex graph.

Lemma 2. *Let G be a biconnected 4-planar graph with positive flexibility, valid orthogonal representation $\mathcal{R}$, and s and t on a common face f. The flex graph $G^\times_\mathcal{R}$ contains a directed cycle C such that $s \in \mathrm{left}(C)$ and $t \in \mathrm{right}(C)$ if one of the following conditions holds.*

(1) $\mathrm{rot}_\mathcal{R}(\pi_f(s,t)) \geq -2$, *$f$ is the outer face and $\pi_f(s,t)$ is not strictly directed from t to s*
(2) $\mathrm{rot}_\mathcal{R}(\pi_f(s,t)) \geq 0$ *and f is the outer face*
(3) $\mathrm{rot}_\mathcal{R}(\pi_f(s,t)) \geq 6$

Proof (Sketch). Assume such a cycle C does not exist. By the duality of cycles and cuts, this implies that there is no directed cut (S,T) in G with $s \in S$ and $t \in T$. Thus for every partition $V = S \dot{\cup} T$ with $s \in S$ and $t \in T$ there is a directed edge with its source in T and its target in S. Iteratively applying this argument yields a path π in G strictly directed from t to s. For each of the conditions (1)–(3), we obtain a contradiction by applying Lemma 1 to the subgraph of G consisting of the strictly directed path π and the path $\pi_f(s,t)$. □

As edges with many bends imply the existence of paths with high rotation, we can use Lemma 2 to successively reduce the number of bends on edges with many bends. Since we only bend along cycles in the flex graph, neither the embedding nor the angles around vertices are changed.

Theorem 2. *Let G be a biconnected 4-planar graph with positive flexibility and valid orthogonal representation. Then G has a valid orthogonal representation with the same planar embedding, the same angles around vertices and at most three bends per edge, except for one edge on the outer face with up to five bends.*

Proof (Sketch). We iteratively bend along cycles in the flex graph to reduce the number of bends on edges with more than three bends. To ensure that the number of bends of an edge does not increase above three once it is below, we set its flexibility down to its current number of bends (but at least 1).

Let $e = \{s,t\}$ be an edge with more than three bends having its negative rotation in an internal face f, i.e., $\mathrm{rot}(e_f) \leq -4$, and assume that $\pi_f(t,s)$ consists of e. As the rotation around the face f is 4, we have $\pi_f(s,t) = 4 - \mathrm{rot}(s_f) - \mathrm{rot}(t_f) - \mathrm{rot}(e_f)$, yielding $\pi_f(s,t) \geq 6$. By Lemma 2 a cycle C separating s from t and thus containing $e^\star$ exists. Bending along this cycle reduces the number of bends on e. With a similar argument, the bends of edges having their negative rotation on the outer face can be reduced to 5. Moreover, if e is an edge with $\mathrm{rot}(e_f) \leq -4$, where f is the outer face, and the boundary of the outer face contains a path with non-negative rotation, the bends of e can be reduced by case (1) of Lemma 2. This is the case if there is another edge e' with $\mathrm{rot}(e'_f) \leq -4$; see Fig. 1(d) where one of the paths π_1 or π_2 must have non-negative rotation. Repeatedly applying this operation yields the theorem. □

If we allow the embedding to be changed slightly, we obtain an even stronger result. Assume the edge e lying on the outer face has five bends. Rerouting e in the opposite direction around the rest of the graph yields a drawing where e has only three bends. Thus, there might be a single edge with up to four bends in the worst case. Note that this result is restricted to biconnected graphs. For general graphs it implies that each block contains at most a single edge with up to four bends. Figure 1(e) illustrates an instance of FlexDraw with linearly many blocks and linearly many edges requiring four bends. We note that increasing the lower bound for the flexibilities from 1 to 2 in the above arguments yields a result similar to the existence of 2-bend drawings by Biedl and Kant [1].

Theorem 2 implies that it is sufficient to consider the flexibility of every edge to be at most 5, or in terms of costs, it is sufficient to store the cost function of

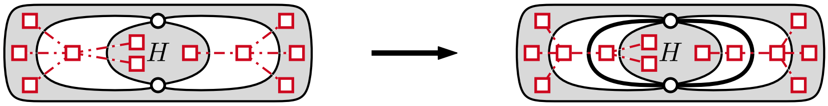

Fig. 2. Adding the safety edges (bold) to G and the effects on the dual graph

an edge only in the interval $[0, 5]$. However, there are two reasons why we need a stronger result. First, we want to compute cost functions of split components and thus we have to limit the number of "bends" they can have (we deal with this in the next section). Second, the cost function of a split component may already be non-convex on the interval $[0, 5]$ [3]. Fortunately, there may be at most a single edge with up to five bends, all remaining edges have at most three bends and thus we only need to consider the interval $[0, 3]$.

4 Flexibility of Split Components and Nice Drawings

Let G be a biconnected 4-planar graph with SPQR-tree $\mathcal{T}$ rooted at some node τ. Recall that we do not require τ to be a Q-node. A node $\mu \neq \tau$ of $\mathcal{T}$ has a unique parent and skel(μ) contains a unique virtual edge $\varepsilon = \{s, t\}$ that is associated with this parent. We call the split-pair $\{s, t\}$ a *principal split pair* and the pertinent graph pert(μ) with respect to the root τ a *principal split component*. The vertices s and t are the *poles* of this split component. Note that an edge (whose Q-node is not τ) is also a principal split component.

Let $\mathcal{R}$ be a valid orthogonal representation of G such that the planar embedding of $\mathcal{R}$ is represented by $\mathcal{T}$ rooted at τ. Consider a principal split component H with respect to the split pair $\{s, t\}$ and let $\mathcal{S}$ be the restriction of $\mathcal{R}$ to H. Note that the poles s and t are on the outer face f of $\mathcal{S}$. We define $\max\{|\operatorname{rot}_{\mathcal{S}}(\pi_f(s, t))|, |\operatorname{rot}_{\mathcal{S}}(\pi_f(t, s))|\}$ to be the *number of bends* of the split component H. With this terminology we can assign a *flexibility* flex(H) to H and we define $\mathcal{R}$ to be *valid* if and only if H has at most flex(H) bends. We say that the graph G has *positive flexibility* if the flexibility of every principal split component is at least 1. Note that this terminology extends the notion of bends and flexibility for edges.

To obtain a result similar to Lemma 2 we need to extend the flex graph such that it respects flexibilities of principal split components. As we cannot deal with principal split components with respect to different roots at the same time, we initially choose an arbitrary Q-node τ to be the root of the SPQR-tree $\mathcal{T}$. We then augment G for each principal split component H with two so-called *safety edges* between the poles; see Fig. 2. As a cycle in the flex graph of the augmented graph crosses H if and only if it crosses these safety edges, suitably orienting them ensures that bending along a cycle in the flex graph does not increase the number of bends of H above flex(H). As we consider only principal split components, the safety edges can be added for all of them at the same time without losing planarity. Denote the resulting augmented graph by G^+ and call the resulting flex graph the *extended flex graph*. As a directed safety edge represents a path with rotation at least 1 along the outer face of its split component, a strictly directed path in G^+ yields a path in G with positive rotation. All remaining arguments from the proof of Lemma 2 can be applied literally, yielding the following lemma.

Lemma 3. *Let G be a biconnected 4-planar graph with positive flexibility, valid orthogonal representation $\mathcal{R}$, and s and t sharing a face f. The extended flex graph contains a directed cycle C such that $s \in \mathrm{left}(C)$ and $t \in \mathrm{right}(C)$, if one of the following conditions holds.*
(1) $\mathrm{rot}_{\mathcal{R}}(\pi_f(s,t)) \geq -2$, *$f$ is the outer face and $\pi_f(s,t)$ is not strictly directed from t to s*
(2) $\mathrm{rot}_{\mathcal{R}}(\pi_f(s,t)) \geq 0$ *and f is the outer face*
(3) $\mathrm{rot}_{\mathcal{R}}(\pi_f(s,t)) \geq 6$

We define a valid orthogonal representation of G to be *nice* if 1) it is tight, 2) every principal split component has at most three bends, and 3) the edge corresponding to the root τ of the SPQR-tree, in case τ is a Q-node, has at most five bends. The following statement extends Theorem 2. Moreover, it obviously extends from FlexDraw to OptimalFlexDraw, i.e., every positive-convex 4-planar graph has an optimal drawing that is nice.

Theorem 3. *Every biconnected 4-planar graph with positive flexibility having a valid orthogonal representation has a valid orthogonal representation with the same planar embedding and the same angles around vertices that is nice with respect to at least one node chosen as root of its SPQR-tree.*

Proof (Sketch). Similar to the proof in Theorem 2 we can use Lemma 3 to reduce the number of bends of split components having their negative rotation in an internal face down to three, while preserving this property once it is achieved by reducing the flexibilities. Similarly, the number of bends of split components with negative rotation in the outer face can be reduced to five. Assume we have an orthogonal representation where each principal split component with more than three bends has its negative rotation on the outer face. If there are two disjoint components of this type, a similar argument as for single edges can be used to reduce the number of bends of one of them. For the remaining nested principal split components of this type we can show that there is no need to reduce the number of bends further, as the drawing is already nice if we reroot the SPQR-tree at the node corresponding to the innermost of these split components. □

5 Optimal Drawings with Variable Planar Embedding

All results presented so far fix the planar embedding of the input graph. In the following we optimize over all embeddings of a biconnected 4-planar graph G. As we only consider positive-convex instances of OptimalFlexDraw, it suffices to consider nice drawings (Theorem 3). Whether a drawing is nice depends on the node chosen as the root for the SPQR-tree $\mathcal{T}$. For a node τ we call an orthogonal representation *τ-optimal* if it is optimal among all representations that are nice with respect to the root τ. We say that it is $(\tau, \mathcal{E})$-*optimal* if it is optimal among all orthogonal representations that are nice with respect to τ and induce the planar embedding $\mathcal{E}$ on $\text{skel}(\tau)$. Computing a $(\tau, \mathcal{E})$-optimal solution for every planar embedding $\mathcal{E}$ of $\text{skel}(\tau)$ obviously yields a τ-optimal orthogonal representation. Moreover, the minimum of all τ-optimal solutions over all nodes of the SPQR-tree yields an overall optimal orthogonal representation. Since G has maximum degree 4, $\text{skel}(\tau)$ has $O(|\text{skel}(\tau)|)$ embeddings (including the choice of an outer face), and hence the sum over all embeddings of all nodes of the SPQR-tree is in $O(n)$. Thus, an algorithm computing a $(\tau, \mathcal{E})$-optimal solution can be used to compute an overall optimal solution by applying it $O(n)$ times.

In the following we show how to compute a $(\tau, \mathcal{E})$-optimal solution efficiently, using a dynamic program computing cost functions of principal split components bottom-up in the SPQR-tree. We start by defining the cost function $\text{cost}_H(\cdot)$ of a principal split component H with poles s and t. Recall that the number of bends of H with respect to an orthogonal representation $\mathcal{S}$ with s and t on the outer face f is defined to be $\max\{|\text{rot}_{\mathcal{S}}(\pi_f(s,t))|, |\text{rot}_{\mathcal{S}}(\pi_f(t,s))|\}$. This implies that there is a lower bound of $\ell_H = \lceil(\deg(s) + \deg(t) - 2)/2\rceil$ bends. For a number of bends $\rho \geq \ell_H$, we define $\text{cost}_H(\rho)$ to be the cost of an orthogonal representation of H that is optimal among all nice representations with ρ bends. For $\rho \in [0, \ell_H)$ we formally set $\text{cost}(\rho) = \text{cost}(\ell_H)$. As we are only interested in nice drawings, it remains to compute $\text{cost}_H(\rho)$ for $\rho \in [\ell_H, 3]$. One of the main results of this section is the following theorem.

Theorem 4. *If the poles of a principal split component do not both have degree 3, then its cost function is convex on the interval* $[0, 3]$.

We prove Theorem 4 later and first assume that it holds. The *base cost* b_H of a principal split component is the minimum of $\text{cost}_H(\cdot)$. Due to Theorem 4 we have $b_H = \text{cost}_H(0)$ except for the single case that $\deg(s) = \deg(t) = 3$, where $\text{cost}_H(\cdot)$ may be non-convex. In this case $b_H = \text{cost}_H(3)$.

In the following we assume that the cost function of every principal split component with respect to the root τ is already computed and show how this can be used to compute a $(\tau, \mathcal{E})$-optimal solution. To this end, we define a flow network on $\text{skel}(\tau)$, similar to Tamassia's flow network [9]. The cost functions computed for the children of τ will be used as cost functions on arcs in the flow network. Since only flow networks with convex costs can be solved efficiently, we have to deal with potentially non-convex cost functions in the case where both poles have degree 3. Our strategy is to simply ignore these subgraphs by contracting them into single vertices. The following lemma justifies this strategy.

Lemma 4. *Let G be a biconnected positive-convex instance of* OptimalFlexDraw *with τ-optimal orthogonal representation $\mathcal{R}$ and let H be a principal split component with non-convex cost function and base cost b_H. Let further G' be the graph obtained from G by contracting H into a single vertex and let $\mathcal{R}'$ be a τ-optimal orthogonal representation of G'. Then* $\text{cost}(\mathcal{R}) = \text{cost}(\mathcal{R}') + b_H$ *holds.*

Now we are ready to define the flow network $N^{\mathcal{E}}$ on skel(τ) with respect to its fixed embedding $\mathcal{E}$. For each vertex v, each virtual edge ε and each face f in skel(τ) the flow network $N^{\mathcal{E}}$ contains the nodes v, ε and f, called *vertex node*, *edge node* and *face node*, respectively. The network $N^{\mathcal{E}}$ contains the arcs (v, f) and (f, v) with capacity 1, called *vertex-face arcs*, if the vertex v and the face f are incident. For every virtual edge ε we add *edge-face arcs* (ε, f) and (f, ε) if f is incident to ε. We use $\mathrm{cost}_H(\cdot) - b_H$ as cost function of the arc (f, ε), where H is the expansion graph of ε. The edge-face arcs (ε, f) in the opposite direction have infinite capacity with 0 cost. Every inner face has a demand of 4, the outer face has a demand of -4. An edge node ε stemming from the edge $\varepsilon = \{s, t\}$ with expansion graph H has demand $\deg_H(s) + \deg_H(t) - 2$, where $\deg_H(v)$ denotes the degree of v in H. The demand of a vertex node v is $4 - \deg_G(v) - \deg_{\mathrm{skel}(\tau)}(v)$.

In the flow network $N^{\mathcal{E}}$, the flow entering a face node f via a vertex-face arc or an edge-face arc is interpreted as the rotation at this vertex or along the path between the poles of its expansion graph, respectively, where incoming flow is positive rotation. Thus, a feasible flow describes the shapes of all expansion graphs and the composition of their representations at vertices. Note that this composition is possible as we can assume them to be tight. Let $b_{H_1}, \dots, b_{H_k}$ be the base costs of the children of τ. We define the *total base costs* of τ to be $b_\tau = \sum_i b_{H_i}$. It can be shown that an optimal flow ϕ in $N^{\mathcal{E}}$ corresponds to a $(\tau, \mathcal{E})$-optimal orthogonal representation $\mathcal{R}$ of G, with costs differing by the total base costs, i.e., $\mathrm{cost}(\mathcal{R}) = \mathrm{cost}(\phi) + b_\tau$. We obtain the following lemma, where $T_{\mathrm{flow}}(\cdot)$ is time necessary to compute an optimal flow.

Lemma 5. *Let G be a biconnected positive-convex instance of* OptimalFlexDraw, *let $\mathcal{T}$ be its SPQR-tree with root τ and let $\mathcal{E}$ be an embedding of* skel(τ). *If the cost function of every principal split component is known, a $(\tau, \mathcal{E})$-optimal solution can be computed in $O(T_{\mathrm{flow}}(|\,\mathrm{skel}(\tau)|))$ time.*

It remains to show that Theorem 4 holds. We make a structural induction over the SPQR-tree. For the leaves it obviously holds as edges are required to have convex costs. For inner nodes we show the following lemma.

Lemma 6. *If Theorem 4 holds for each principal split component corresponding to a child of the node μ in the SPQR-tree, then it also holds for* pert(μ).

Proof (Sketch). In an orthogonal representation $\mathcal{S}$ of $H = \mathrm{pert}(\mu)$, the number of bends ρ is determined by the rotation along one of the two paths $\pi(s, t)$ or $\pi(t, s)$. We define the *partial cost function* $\mathrm{cost}_H^{\mathcal{E}}(\cdot)$ with respect to the embedding $\mathcal{E}$ of skel(μ) to be the smallest possible cost of an orthogonal representation inducing the planar embedding $\mathcal{E}$ on skel(μ) with ρ bends such that $\pi_f(s, t)$ determines the number of bends. We show how to compute $\mathrm{cost}_H^{\mathcal{E}}(\cdot)$ using a flow network similar to $N^{\mathcal{E}}$. It can be shown that these partial cost functions are convex, and that their minimum $\mathrm{cost}_H(\cdot)$ defined by $\mathrm{cost}_H(\rho) = \min_{\mathcal{E}} \mathrm{cost}_H^{\mathcal{E}}(\rho)$ is convex.

We define $N^{\mathcal{E}}$ as before with two changes. First, the parent edge plays a special role as it should not occur in the resulting orthogonal representation. Removing some arcs and adjusting the demands accordingly yields a flow network such that an optimal flow corresponds to an optimal orthogonal representation. Second, the flow network is parameterized as follows. The incoming flows at the two face-nodes corresponding to the faces incident to the parent edge are equal to the rotations along the paths $\pi(s,t)$ and $\pi(t,s)$ in a corresponding orthogonal representation. We parameterize $N^{\mathcal{E}}$ with respect to these two faces. It can then be shown that the cost function of the flow and the partial cost function of H coincide on the interval $[\ell_H, 3]$ up to the total base cost. Thus, convexity for the partial cost function follows from the convexity of the cost function of a parametrized flow network if $\mathrm{cost}_{N^{\mathcal{E}}}(\rho)$ is minimal for $\rho = \ell_H$; see Theorem 1. We note that this is not obvious and not even true if $\deg(s) = \deg(t) = 3$. However, it is true for all other cases. Moreover, we can show that the minimum over all partial cost functions is convex on the interval $[\ell_H, 3]$. This is again not obvious and not even true for a larger interval [3]. □

The proof of Lemma 6 yields an algorithm computing a $(\tau, \mathcal{E})$-optimal solution bottom-up in the SPQR-tree. In each node μ a constant number of optimal flows in a network of size $|\operatorname{skel}(\mu)|$ has to be computed, consuming overall $O(T_{\text{flow}}(n))$ time. Applying this algorithm $O(n)$ times yields an optimal drawing.

Theorem 5. OptimalFlexDraw *can be solved in* $O(n \cdot T_{\text{flow}}(n))$ *time for positive-convex biconnected instances.*

We can extend our algorithm to the case where G contains cutvertices (an extensive description is in the appendix), yielding the following theorem.

Theorem 6. OptimalFlexDraw *can be solved in* $O(n^2 \cdot T_{\text{flow}}(n))$ *time for positive-convex instances.*

References

1. Biedl, T., Kant, G.: A Better Heuristic for Orthogonal Graph Drawings. Comput. Geom. 9(3), 159–180 (1998)
2. Bläsius, T., Krug, M., Rutter, I., Wagner, D.: Orthogonal graph drawing with flexibility constraints. Algorithmica (2012), doi:10.1007/s00453-012-9705-8
3. Bläsius, T., Rutter, I., Wagner, D.: Optimal orthogonal graph drawing with convex bend costs. CoRR abs/1204.4997 (2012), `http://arxiv.org/abs/1204.4997`
4. Di Battista, G., Liotta, G., Vargiu, F.: Spirality and Optimal Orthogonal Drawings. SIAM J. Comput. 27(6), 1764–1811 (1998)
5. Di Battista, G., Tamassia, R.: On-Line Maintenance of Triconnected Components with SPQR-Trees. Algorithmica 15(4), 302–318 (1996)
6. Di Battista, G., Tamassia, R.: On-Line Planarity Testing. SIAM J. Comput. 25(5), 956–997 (1996)
7. Garg, A., Tamassia, R.: On the Computational Complexity of Upward and Rectilinear Planarity Testing. SIAM J. Comput. 31(2), 601–625 (2001)
8. Gutwenger, C., Mutzel, P.: A Linear Time Implementation of SPQR-Trees. In: Marks, J. (ed.) GD 2000. LNCS, vol. 1984, pp. 77–90. Springer, Heidelberg (2001)
9. Tamassia, R.: On Embedding a Graph in the Grid with the Minimum Number of Bends. SIAM J. Comput. 16(3), 421–444 (1987)