Orthogonal Layout with Optimal Face Complexity[☆]

Md. Jawaherul Alam

Department of Computer Science, University of California, Irvine, USA

Stephen G. Kobourov

Department of Computer Science, University of Arizona, USA

Debajyoti Mondal[*]

David R. Cheriton School of Computer Science, University of Waterloo, Canada

Abstract

We study a problem motivated by rectilinear schematization of geographic maps. Given a biconnected plane graph G and an integer $k \geq 0$, does G have a strict-orthogonal drawing (i.e., an orthogonal drawing without edge bends) with at most k reflex angles per face? For $k = 0$, the problem is equivalent to realizing each face as a rectangle. We prove that the strict-orthogonal drawability problem for arbitrary reflex complexity k can be reduced to a graph matching or a network flow problem. Consequently, we obtain an $\widetilde{O}(n^{10/7}k^{1/7})$-time algorithm to decide strict-orthogonal drawability, where $\widetilde{O}(r)$ denotes $O(r\ log^c r)$, for some constant c. In contrast, if the embedding is not fixed, we prove that it is NP-complete to decide whether a planar graph admits a strict-orthogonal drawing with reflex face complexity 4.

Keywords: Graph Drawing, Orthogonal Drawing, Face Complexity.

1. Introduction

Map schematization is a problem of interest in geography, cartography, information visualization and computational geometry. Rectangular and rectilinear schematizations have been studied for over 80 years; see the comprehensive survey of Tobler [24]. While rectangular schematizations sometimes must distort the topology of the map (e.g., no four mutually neighboring countries can be represented by contact of rectangles), rectilinear schematizations can preserve

[☆]A preliminary version of the paper appeared in the Forty-First International Conference on Current Trends in Theory and Practice of Computer Science (SOFSEM 2016) [1].

[*]Corresponding author

Email addresses: `alamm1@uci.edu` (Md. Jawaherul Alam), `kobourov@cs.arizona.edu` (Stephen G. Kobourov), `dmondal@uwatreloo.ca` (Debajyoti Mondal)

the topology, at the expense of more complicated country shapes. We consider the problem of rectangular schematization where the "complexity" of each country (as defined by the number of reflex corners) is minimized. We also consider the case where different countries are allowed to have different complexities. We describe efficient algorithms for both of these scenarios.

An *orthogonal drawing* of a planar graph $G = (V, E)$ in $\mathbb{R}^2$ is a planar drawing of G such that each vertex $v \in V$ is drawn as a point and each edge $(u, v) \in E$ is drawn as a rectilinear (axis-aligned) path between the points that correspond to u and v. A *t-bend orthogonal drawing* of G is an orthogonal drawing of G, where each edge is drawn as an orthogonal polyline with at most t bends. An orthogonal drawing is *strict* if it does not contain any bends, i.e., it is a 0-bend orthogonal drawing. In the literature such a drawing is also referred to as *bendless* or *no-bend* orthogonal drawing [22]. If G is a *plane graph* (i.e., a planar graph with a fixed planar embedding), then an orthogonal drawing of G is additionally constrained to respect the given planar embedding. The *reflex face complexity* of an orthogonal drawing Γ is the smallest integer k such that each inner face of Γ contains at most k reflex angles, and the outer face of Γ contains at most $k + 4$ reflex angles. Thus in an orthogonal drawing of G with reflex face complexity k, each face of G is drawn as an orthogonal polygon with at most $2k + 4$ sides. Figs. 1(a)-(c) show a graph G and two strict-orthogonal drawings of G.

From technical drawings and wiring schematics to transportation network layouts, orthogonal drawing (or layout) is one of the standard types of visualization for planar graphs [8, 15, 21] and is also supported by most network layout systems (e.g., yEd [26], graphviz [9], and OGDF [5]). Early work on orthogonal layouts was done by Valiant [25] and Leiserson [18] in the context of VLSI design. The input graphs are assumed to be planar and with maximum-degree four, although models incorporating higher degree graphs were introduced later by Tamassia [23] and Fößmeier and Kaufmann [11].

1.1. Optimization Goals and Challenges

The number of reflex corners per face and the number of bends per edge are two important parameters in an orthogonal drawing, and a good drawing usually minimizes these parameters. Note that these two parameters are important not only from the point of view of the complexity of a VLSI layout or a floor-plan, but also because they influence the readability and aesthetics of the drawing. Recently, Keiffer et al. [16] proposed several design principles based on human subject studies with orthogonal drawings, and developed an algorithm that incorporates these principles while computing the drawing. Specifically, the results showed that edge bends are often correlated with preferences and ranking. Minimizing the total number of bends over all possible embeddings of the input planar graph is NP-hard [12], however, for maximum-degree-4 plane graphs, Tamassia [23] proposed a maximum-flow approach to solve the problem in $O(n^{7/4}\sqrt{\log n})$-time. Later, Cornelsen and Karrenbauer [6] improved the maximum-flow approach to $O(n^{3/2})$. Although these algorithms can be adapted to bound the number of bends per edge, there exist more specialized algorithms

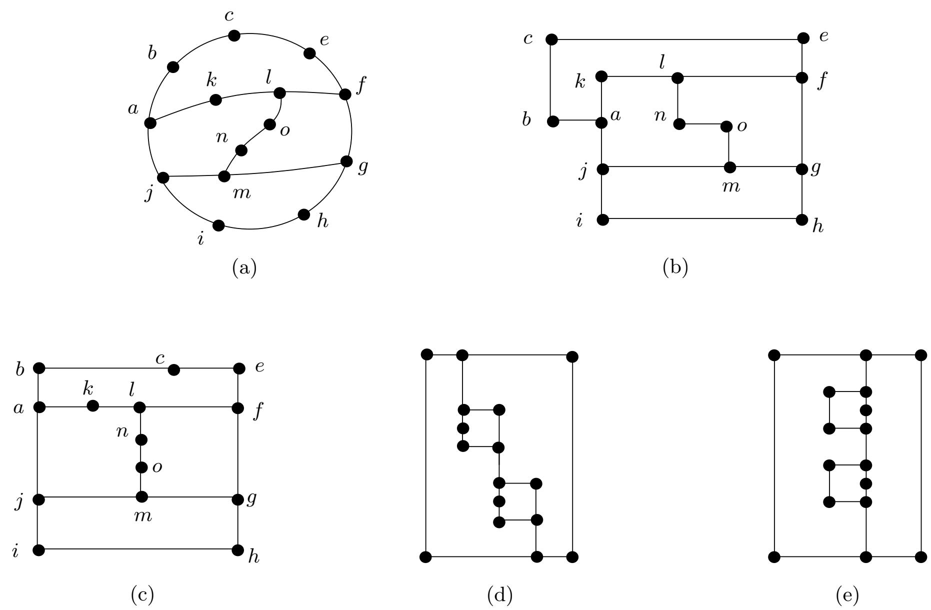


Figure 1: (a) A plane graph G. (b) A strict-orthogonal drawing of G with reflex face complexity 1. (c) A rectangular drawing of G. (d)–(e) Two strict-orthogonal drawings (0-bend drawings) of the same graph with different reflex face complexities.

for such optimizations. For example, Bläsius et al. [3, 4] gave efficient algorithms to bound the number of bends per edge, which can also optimize any convex cost associated with the edges of the input graph, even in the variable embedding setting for some specific cost functions.

Note that minimization of the number of total bends, or the number of bends per edge cannot bound the reflex face complexity, see Figs. 1(d)–(e), but a drawing with reflex face complexity k ensures that the number of bends per edge is at most $2k + 4$. Given a plane graph G with four prescribed corner vertices, Miura *et al.* [20] showed how to decide whether G admits a strict-orthogonal drawing with reflex face complexity 0 (also known as *rectangular drawings*, as shown in Fig. 1(c)), that respects the given corners. They reduced the problem of rectangular drawing to the problem of finding a perfect matching in some graph, which leads to an $O(n^{1.5}/\log n)$-time algorithm. If the four corner vertices are not given, then a trivial solution is to try all possible options for the corner vertices. A variant of Tamassia's [23] flow-based approach can solve this problem in $O(n \log^2 n)$ time, even when the corners are not given in the input (we refer the reader to Section 3 for the details).

The flow-based approach of Tamassia [23] can be modified to decide strict-orthogonal drawability for arbitrary reflex complexity k by solving a maximum-flow problem in $\widetilde{O}(n^{10/7}k^{1/7})$ time, as described in Section 3. Other variations of Tamassia's formulation [3, 4, 6] can also be adapted to decide strict-orthogonal drawability with a given reflex face complexity. Thus an interesting question is whether the matching-based approach of Miura *et al.*'s [20] can also be generalized to decide orthogonal drawability with reflex face complexity k.

1.2. Our Contributions

We study the problem of orthogonal drawing of a planar graph with a given reflex face complexity k. Note that since every vertex in an orthogonal drawing has degree at most 4, we consider only max-degree-4 graphs in this paper. In the fixed embedding setting, we reduce the problem of computing strict-orthogonal drawing with any given reflex face complexity k (if such a drawing exists) to a graph matching or a network flow problem. Furthermore, given the nonnegative integers $k_0, k_1, \ldots, k_r$ for the faces $f_0, f_1, \ldots, f_r$ of G, our algorithm can compute a strict-orthogonal drawing of G, with at most k_i reflex corners in each face f_i, $i \in \{0, 1, \ldots, r\}$. For example, one can specify $k_i = k$ for each inner face f_i, and $k_0 = 4$ for the outer face f_0 to compute a complexity-k tessellation of a rectangle.

For biconnected graphs, our matching-based algorithm runs in $\widetilde{O}((nk)^{10/7})$ time (see Section 2), which is slower than the $\widetilde{O}(n^{10/7}k^{1/7})$-time flow-based algorithm (see Section 3). Hence the matching-based approach is mostly of theoretical interest. Our algorithm generalizes to simply connected graphs, and furthermore, it can be extended to compute (non-strict) orthogonal drawings with at most t_i bends on each edge e_i, for some nonnegative integer t_i. However, these generalizations lead to a slower running time.

Finally, we show that if the embedding of the planar graph G is not given, then deciding whether G has a strict-orthogonal drawing with a given reflex face complexity k is NP-complete, even when $k = 4$.

2. Strict-Orthogonal Drawing Algorithms for Plane Graphs

In this section we describe our algorithm for deciding strict-orthogonal drawability of planar graphs with a given reflex face complexity, and discuss some generalizations. We begin with a preliminary result showing that to compute a strict-orthogonal drawing it suffices to specify the angles between pairs of consecutive edges around each vertex (Section 2.1). We then describe our matching-based algorithm (Section 2.2), where we restrict the input to be a biconnected planar graph. Finally, we relax the connectivity constraint and discuss further generalizations of our algorithm (Section 2.3).

2.1. Orthogonal Drawing using Angle Assignment

Tamassia [23] showed that an orthogonal drawing Γ of a biconnected plane graph G can be described by augmenting the embedding of G with the angles at the bends (*bend angles*) and the angles between pairs of consecutive edges around the vertices of G (*vertex angles*). For strict-orthogonal drawings (no bends), we only consider vertex angles. Consider an *angle assignment* of G, where each vertex angle is assigned an element from $\{\pi/2, \pi, 3\pi/2\}$. Although an angle assignment of G does not specify edge lengths, it can precisely describe the shape of Γ. Given an angle assignment Φ, one can test if Φ corresponds to a strict-orthogonal drawing by Lemma 1, which is implied from [23]:

Lemma 1. *An angle assignment Φ for a plane graph G corresponds to a strict-orthogonal drawing of G if and only if Φ satisfies the following conditions (P_1–P_2):*

(P_1) *The sum of the assigned angles around each vertex v in G is 2π.*

(P_2) *the total assigned angle of every inner (respectively, outer) face f is $(\gamma-2)\pi$ (respectively, $(\gamma+2)\pi$), where γ is the number of vertices on the boundary of f.*

Given an angle assignment Φ satisfying (P_1–P_2), a strict-orthogonal drawing of G (i.e., the exact coordinates for the vertices) can be computed in linear time.

2.2. Bipartite Graph Matching Formulation

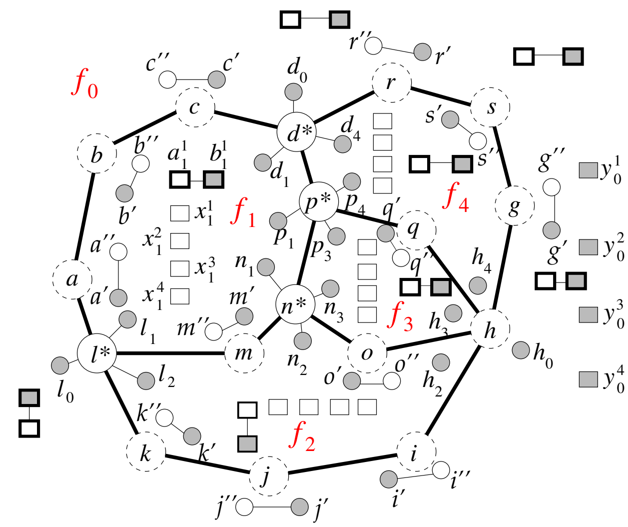
In this section we assume that G is biconnected, and reduce the orthogonal drawability problem to the problem of finding a perfect matching in a bipartite graph. We construct a bipartite graph $B(G)$ so that one can compute a strict-orthogonal drawing of G with reflex face complexity k from a perfect matching of $B(G)$, and vice versa. Although our result generalizes the rectangular drawing algorithm by Miura *et al.* [20], the bipartite graph we construct is quite different from the one in [20] and it gives the option of having reflex corners in a face.

2.2.1. Construction of $B(G)$:

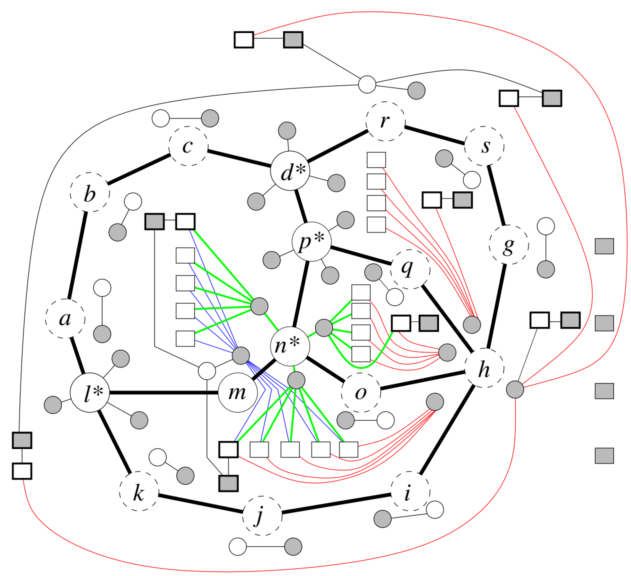
Let f_0 be the outer face and $f_1, \ldots, f_r$ be the inner faces of G; see Fig. 2(a). Let $k_0(\geq 4), k_1, \ldots, k_r$ be a set of nonnegative integers, where the face f_j, $0 \leq j \leq r$, is allowed to have at most k_j reflex corners.

For each inner face f_i, $i \in \{1, \ldots, r\}$ of G we have four x-vertices x_i^1, x_i^2, x_i^3, x_i^4 in $B(G)$, as shown with white squares with thin boundaries. These vertices will correspond to four $\pi/2$ angles in f_i. We also have k_i pairs of a and b-vertices a_i^1, $b_i^1, \ldots, a_i^{k_i}$, $b_i^{k_i}$ associated with f_i, as shown with white and gray squares with bold boundaries. For each $j \in \{1, \ldots, k_i\}$, there is an edge (a_i^j, b_i^j). Later, every a-vertex will correspond to a $\pi/2$ angle, and every b-vertex will correspond to a $3\pi/2$ angle in f_i. In each internal face f_i, there are only k_i pairs of a and b-vertices, which will bound the number of reflex corners of f_i in the final drawing. Observe that by Condition (P_2) of Lemma 1, each internal face of G has exactly four $\pi/2$ angles more than its $3\pi/2$ angles, and hence we have four more white squares (i.e., x and b-vertices) than gray squares (i.e., b-vertices). Similarly, the outer face f_0 must contain four $3\pi/2$ angles more than its $\pi/2$ angles. Thus for the face f_0, we have four vertices y_0^1, y_0^2, y_0^3 and y_0^4 representing $3\pi/2$ angles, and $p = k_0 - 4$ pairs of vertices a_0^1, $b_0^1, \ldots, a_0^p$, b_0^p. Call the x- and a-vertices the *convex face-vertices* and the y- and b-vertices the *reflex face-vertices*.

In addition to the face-vertices above, $B(G)$ also has boundary-vertices that correspond to the vertices of G. For each degree-4 vertex v in G, let f_i, f_j, f_k,



(a)



(b)

Figure 2: (a) A plane graph G (induced by the bold edges), and the construction of $B(G)$ with $k_0 = 4$, $k_1 = k_2 = k_3 = k_4 = 1$, where only a few edges of $B(G)$ are shown. Note that the bold edges (i.e., the edges of G) and the dashed vertices do not belong to $B(G)$. (b) The remaining edges in $B(G)$: the edges shown are the ones incident to the convex boundary vertices for a degree-4 (red), a degree-3 (green), a degree-2 (blue) vertices and the ones incident to reflex boundary vertices for two degree-2 vertices (black).

f_l be the four faces incident to v. For each $\lambda \in \{i, j, k, l\}$, $B(G)$ has a vertex v_λ, which is adjacent to all the convex face-vertices associated with f_λ; see vertex h in Fig. 2(b). We refer to these vertices as *convex boundary-vertices*. Each of these convex boundary-vertices will choose a convex face-vertex ensuring four $\pi/2$ angles around v. For each degree-3 vertex v incident to the faces f_i, f_j, f_k, $B(G)$ has three vertices v_i, v_j, v_k, which are adjacent to all the convex face-vertices of their corresponding faces. We also have an additional vertex v^* in $B(G)$, which is a common neighbor for v_i, v_j, v_k; see vertex n^* in Fig. 2(b). Again we refer to these vertices v_i, v_j, v_k as *convex boundary-vertices*, and the vertex v^* as the *central-vertex*. Intuitively, v^* will match with one of its incident vertices leaving two vertices among $\{v_i, v_j, v_k\}$, which will choose two $\pi/2$ angles around v. Finally, if v is a degree-2 vertex incident to the faces f_i and f_j, then we have two vertices v' and v'' in $B(G)$ that are adjacent to each other. We call v' a *convex boundary-vertex* (shown as gray circle), and v'' a *reflex boundary-vertex* (shown as white circle). The vertex v' is adjacent to all the convex face-vertices associated with f_i and f_j, and the vertex v'' is adjacent to all the reflex vertices associated with f_i and f_j; see vertex m in Fig. 2(b). Note that degree-3 and degree-4 vertices of G do not have any associated reflex boundary-vertices in $B(G)$, since they cannot induce $3\pi/2$ angles in an orthogonal drawing; see Lemma 1, Condition (P_1).

This completes the construction of $B(G)$, which is indeed a bipartite graph, as shown by coloring the vertices gray and white in Fig. 2(b).

2.2.2. Reduction:

The following lemma reduces our problem to the problem of finding a perfect matching in some corresponding graph.

Lemma 2. *There is a perfect matching in $B(G)$ if and only if G has a strict-orthogonal drawing, where each face f_i contains at most k_i reflex corners.*

Proof: Assume that $B(G)$ has a perfect matching M; see Figs. 3(a)–(b). From this matching, we compute an angle assignment Φ for G from the set $\{\pi/2, \pi, 3\pi/2\}$ so that Φ satisfies Conditions (P_1–P_2) of Lemma 1.

Consider an arbitrary face f_i of G. We assign an angle inside f_i (at some vertex v) the value $\pi/2$ if the corresponding boundary-vertex in $B(G)$ is matched to some convex face-vertex of f_i. For example, the convex boundary-vertices associated with the vertices b and h in Fig. 3(b) are determining $\pi/2$ angles around b and h in Fig. 3(c). Similarly, a $3\pi/2$ angle is assigned to v when its corresponding boundary-vertex in $B(G)$ is matched with a reflex face-vertex for f_i, e.g., see vertex m in Fig. 3(b). Otherwise, the boundary-vertex is either matched with some central-vertex, or another boundary vertex (e.g., see vertex c). In both cases we assign the corresponding angle the value π.

Note that the above rules may lead to a conflict at some degree-2 vertex, when it has both convex and reflex boundary-vertices matched to the convex and reflex face-vertices of the same face. For example, the vertex q in Fig. 3(b) has its boundary vertices matched with the face-vertices in the same face f_3.

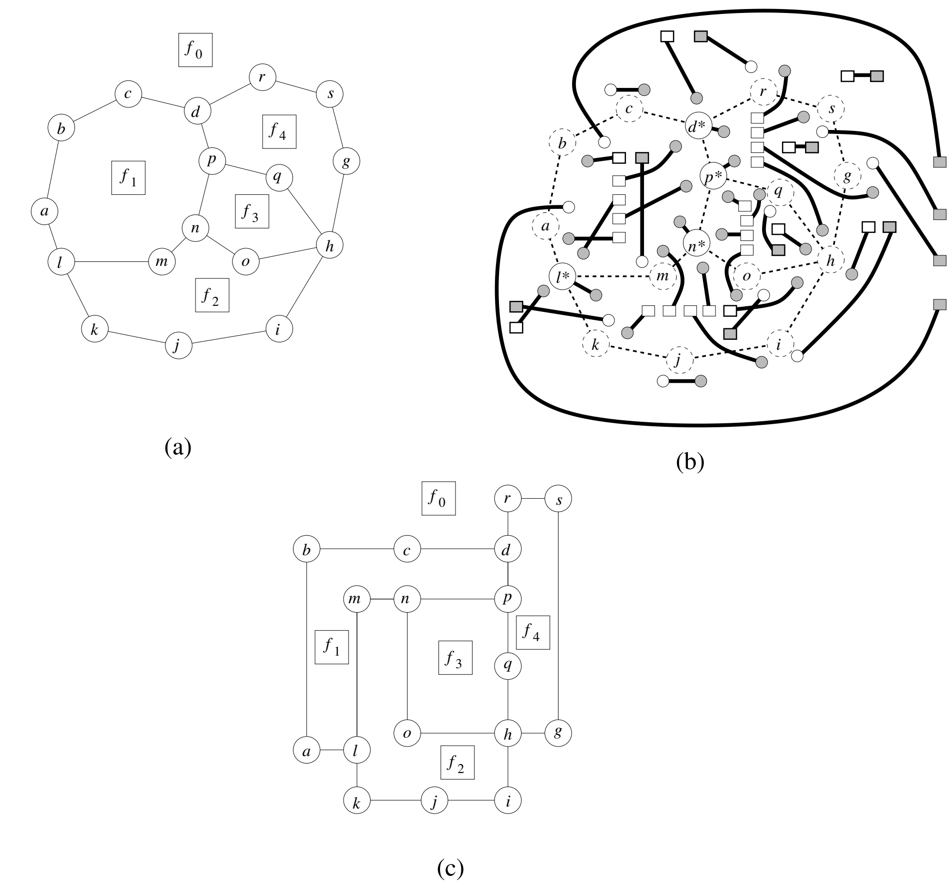


Figure 3: (a) A biconnected plane graph G with maximum degree four, (b) a perfect matching in $B(G)$, and (c) a strict-orthogonal drawing of G with $k_0 = 4$ and $k_1 = k_2 = k_3 = k_4 = 1$.

In such a case we assign the angle at v a value of π (inside the corresponding face). Since M is a perfect matching, the construction of $B(G)$ implies that each inner face has exactly four more $\pi/2$ angles than $3\pi/2$ angles. Similarly, the outer face f_0 contains exactly four more $3\pi/2$ angles than $\pi/2$ angles. Thus Condition (P_2) of Lemma 1 is satisfied for each face of G.

Consider now the assignment of angles around each vertex v of G. If $\deg(v) = 4$, then all its four convex boundary-vertices are matched to some convex face-vertices, and hence it has exactly four $\pi/2$ angles. If $\deg(v) = 3$, then exactly one of its three convex boundary-vertices is matched with v^*, and hence it has two $\pi/2$ angles and one π angle. Finally, if $\deg(v) = 2$, then it either has two π angles (because v' and v'' are either matched to each other or to the face-vertices in the same face), or it receives exactly one $\pi/2$ angle and exactly one $3\pi/2$ angle. Thus the sum of angles around each vertex is 2π, satisfying Condition (P_1) of Lemma 1. By Lemma 1, this angle assignment gives an orthogonal drawing of

G. Since each face f_i can have at most k_i reflex boundary-vertices matched to its k_i reflex face-vertices, the number of reflex corners in the drawing of f_i is at most k_i; see Fig. 3(c).

Conversely, if G has a strict-orthogonal drawing Γ, where each face f_i of G has at most k_i reflex corners, then Γ gives a perfect matching M in G, as follows. For each face f_i of G, traverse around its drawing in Γ, and for each $\pi/2$ (respectively, $3\pi/2$) angle, match the corresponding boundary-vertex to a convex (respectively, reflex) face-vertex of f_i. There are always sufficiently many face-vertices, since each inner face f_i is associated with k_i pairs of convex and reflex face-vertices, and the outer face f_0 has exactly $p = k_0 - 4$ such pairs. It is straightforward to match face-vertices with boundary vertices such that the unmatched face-vertices remain in pairs. Hence we can afterwards choose the edges between the unmatched pairs of face-vertices in M. For each degree-2 vertex with two π angles, we take the edge between its boundary-vertices in M. Finally, for each degree-3 vertex v, we match the boundary vertex corresponding to the π angle of v with v^*. □

2.2.3. Time Complexity:

The number of vertices $|V|$ in $B(G)$ is $O(nk)$, where $k = \max_i\{k_i\}$. Since there are $O(n)$ boundary-vertices, and for each of the $O(n)$ faces there are $O(k)$ face-vertices, the number of edges $|E|$ in $B(G)$ is again $O(nk)$. In the preliminary version of this paper [1], we used the Hopcroft-Karp algorithm [14] to test for the existence of a perfect matching in $B(G)$ in $O(\sqrt{|V|}|E|) = O(\sqrt{nk} \times nk) = O((nk)^{1.5})$ time. However, based on the best known time-complexity for computing a maximum bipartite matching [19], a perfect matching in $B(G)$ can be computed in $O(|E|^{10/7}) = O((nk)^{10/7})$ time, which dominates the running time of our algorithm. Note that this matching-based algorithm is slower than the $\widetilde{O}(n^{10/7}k^{1/7})$-time flow-based algorithm described in Section 3, and hence our matching-based approach is mostly of theoretical interest.

2.3. Generalizations

In this section we show how we can relax the biconnectivity constraint and allow the edges to have bends while drawing strict-orthogonal drawings.

2.3.1. Drawings for Simply Connected Graphs

The algorithm in Section 2.2 works when the input graph is biconnected. If the input graph G is not biconnected, then we can transform the graph in linear time to a biconnected graph G' such that G admits a strict-orthogonal drawing with reflex face complexity k if and only if G' admits a strict-orthogonal drawing with some prescribed bound on the face complexities. We compute G' following Steps 1–3.

Step 1 (Process degree-one vertices): For each degree-one vertex v, we construct a cycle $C = (v, v_1, v_2, v_3)$, assign a reflex face complexity 0 inside and add 3 to the complexity outside of C. The resulting graph now does not contain any degree-one vertex, and such graphs are processed in Step 2.

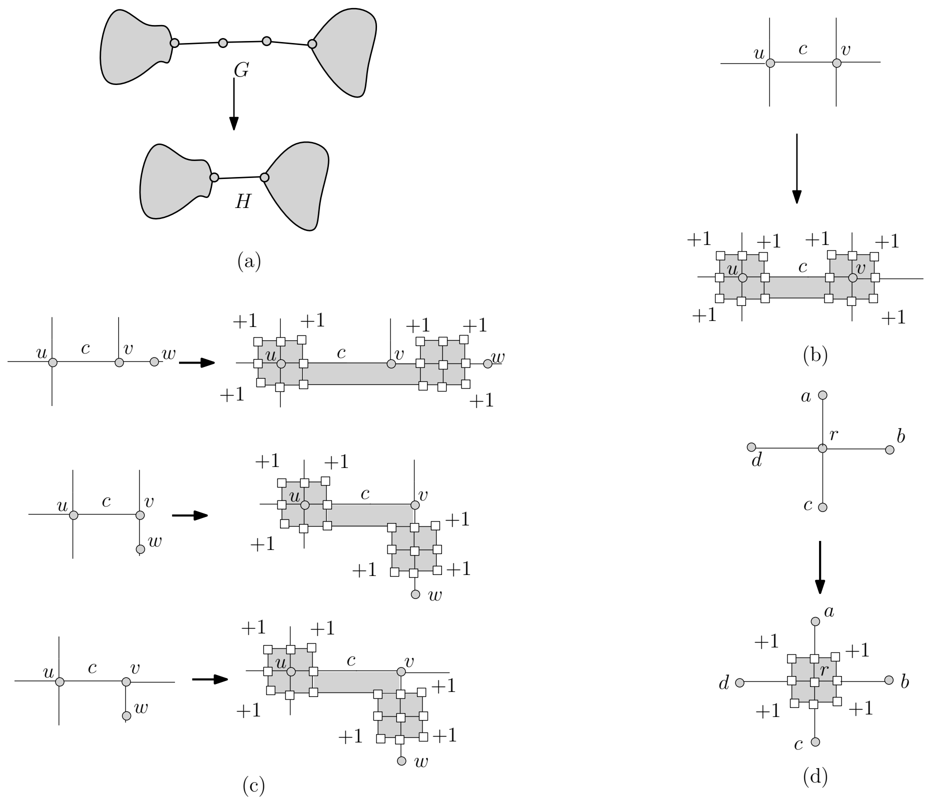


Figure 4: Generalization for simply connected graphs.

Step 2 (Process cut edges): G now does not contain any degree-one vertex. We first replace every sequence of cut edges by a single edge, as shown in Fig. 4(a). Let the resulting drawing be H. It is straightforward to verify that such a modification does not destroy the equivalence of G and H, i.e., any strict orthogonal drawing of H respecting the prescribed bound on face complexity can be modified to have a valid drawing for G, and vice versa.

Assume now that we have a cut edge $c = (u, v)$ in H. Since we replaced every sequence of cut edges in G by a single edge, we have $\deg(u), \deg(v) > 2$ in H. Let the components attached to u and v be C_u and C_v, respectively. We now transform H to a graph H' such that the corresponding components in H' can no longer be disconnected by deleting a single edge, and the transformation preserves the equivalence between H and H'. We describe the transformation corresponding to v distinguishing the following two cases. The modification around u can be carried out in a similar way.

Case A ($\deg(v) = 4$): This transformation is illustrated in Fig. 4(b). Observe that v is enclosed by a cycle of eight vertices, which increases the face complexity of the resulting graph outside the cycle. We can verify from the construction that any strict-orthogonal drawing of H' can be modified to obtain a valid strict-orthogonal drawing of H. All gray shaded faces are assigned reflex face complexity 0, and the reflex face complexities of the adjacent exterior faces have been increased accordingly.

Case B ($\deg(v) = 3$)**:** This transformation is illustrated in Fig. 4(c), which is slightly different than Case A. Observe that this construction allows enough flexibility to freely choose the orientations of the three neighbors around v (preserving the input embedding). For any choice of orientations, the increase in face complexities is consistent, as shown in Fig. 4(c).

Step 3 (Process cut vertices): At this stage H' is 2-edge connected, but it may contain cut vertices. Let r be a cut vertex in H'. If the $\deg(r) \leq 3$, then r must be adjacent to some cut edge, which contradicts that H' is 2-edge connected. We may thus assume that $\deg(r) = 4$. Let a, b, c, d be the neighbors of r in clockwise order. We enclose r by a cycle of eight vertices, as illustrated in Fig. 4(d). All gray shaded faces are assigned reflex face complexity 0, and the reflex face complexities of the adjacent exterior faces have been increased accordingly. We now claim that this transformation does not introduce new cut edges. Let v_a, v_b, v_c, v_d be the division vertices on the edges $(a, r), (b, r), (c, r), (d, r)$, respectively. If (v_q, q), where $q \in \{a, b, c, d\}$ is a cut edge in the resulting graph, then (q, r) must be a cut edge in H', which contradicts that H' is 2-edge connected.

Consequently, after we process all cut vertices, we obtain the required biconnected graph G'.

2.3.2. General Orthogonal Drawing with a Given Face-Complexity

Here we extend our algorithm to general (non-strict) orthogonal drawings. Note that each bend in an orthogonal drawing can be thought of as a degree-2 vertex on some edge in the graph (e.g., a subdivision of an edge). The following lemma is a straightforward consequence of this observation.

Lemma 3. *Let G be a biconnected plane graph with edges $e_1, \ldots, e_m$ and faces $f_0, f_1, \ldots, f_r$. Consider the sets of non-negative integers $t_1, \ldots, t_m$ and $k_0(\geq 4)$, $k_1, \ldots, k_r$. Let G_t be a graph obtained from G by subdividing each edge e_i exactly t_i times. Then G has an orthogonal drawing, where each edge e_i has at most t_i bends and each face f_i has at most k_i reflex corners if and only if G_t has a strict-orthogonal drawing where each face f_i has at most k_i reflex corners.*

We may now use Lemma 3 to find a polynomial-time algorithm for orthogonal drawings that simultaneously bounds the reflex face complexity and the number of bends per edge. Our goal in this paper is to bound the reflex face complexity and we leave the task of designing fast algorithms optimizing multiple objectives as a future work. There exists specialized algorithms for bounding the number of bends per edge or for optimizing any convex cost associated with the edges of the input graph, even in the variable embedding setting for some specific cost functions [4, 3].

The following theorem summarizes the main result of this section.

Theorem 1. *Let G be an n-vertex plane graph with edges $e_1, \ldots, e_m$ and faces f_0, $f_1, \ldots, f_r$. Given the sets of non-negative integers $t_1, \ldots, t_m$ and $k_0(\geq 4)$, $k_1, \ldots, k_r$, one can decide in polynomial time whether G has a strict-orthogonal drawing, where each edge e_i has at most t_i bends and each face f_i has at most k_i reflex corners. Furthermore, such a drawing (if exists) can be computed in polynomial time.*

3. Strict-Orthogonal Drawings via Network Flow

Here we briefly review the network-flow formulations by Tamassia [23] for computing minimum-bend orthogonal drawings of plane graphs. We then describe how this algorithm can be modified to compute drawings with bounded reflex face complexities.

Given a biconnected plane graph G, the corresponding Tamassia's network H contains a set of *boundary-vertices*, V_R, and a set of *face-vertices*, V_F; see Figs. 5(a)–(b). The set V_R of boundary vertices corresponds to the original vertices of H, and the set V_F of face vertices corresponds to the faces of H. The edges of H are the bidirectional edges of the dual graph of G (dashed edges in Fig. 5(b), called *dual edges*) and the edges from each boundary-vertex to its incident face-vertices (solid edges). Each vertex $v \in V_R$ is a source with a production of $4 - \deg(v)$ units, where $\deg(v)$ is the degree of the corresponding vertex in G. The production or consumption of each face-vertex $f \in V_F$ is either $4 - \deg(f)$ units (for inner faces) or $-4 - \deg(f)$ (for the outer face), where $\deg(f)$ is the length of the corresponding face in G. The cost of an edge is 1 unit if it connects two face-vertices, and 0 otherwise. A min-cost flow in this network corresponds to an orthogonal drawing of G, as follows. A flow of $t \in \{0, 1, 2, 3\}$ units from a boundary-vertex to a face-vertex determines a $(t+1)\pi/2$ assignment to the corresponding angle in G. A flow of t units through some dual edge (dashed edge) corresponds to t bends in the corresponding edge of G; see Fig. 5(c). Using this network, Tamassia [23] gave an $O(n^2 \log n)$-time algorithm for orthogonal drawing with minimum number of bends. Cornelsen and Karrenbauer [6] used the same network but improved the running time to $O(n^{1.5})$ with a faster min-cost flow algorithm for this planar network.

One can modify the above network to solve the problem of orthogonal drawings with bounded reflex face complexities as follows; see Fig. 5(d). Delete the dual edges, i.e., dashed edges of H. For each face-vertex v_f in H, add a new vertex v'_f (unfilled red vertices) in H. For each edge (v_b, v_f) in H, with a degree-2 boundary vertex v_b, add the edge (v_b, v'_f). Add the edges (v'_f, v_f) and call the resulting network H'; see Fig. 5(d). Note that this network does not have costs on the edges. Also note that only degree-two vertices can contribute to $3\pi/2$ angles in the drawing. Place a capacity upper bound of 1 unit on each edge that is incident to some degree-two boundary-vertex v_b. Consequently, a $3\pi/2$ angle at v_b inside some face f corresponds to one unit of flow from v_b to v_f and

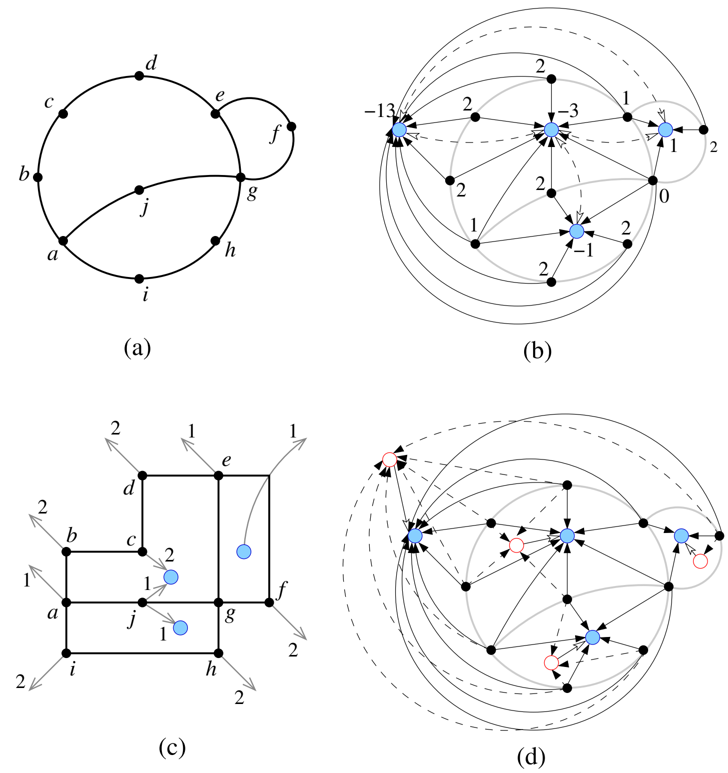


Figure 5: (a) A plane graph G, (b) construction of the flow-network H from G by Tamassia, where V_R corresponds to the vertices of G, and V_F corresponds of the faces of G. (c) An orthogonal drawing of G and the corresponding flow, (d) modification of the network by Tamassia to solve the problem of orthogonal drawing with bounded reflex complexity for the faces.

one unit of flow through v_b, v'_f, v_f. Finally, add a capacity upper bound of k_f on (v'_f, v_f), where k_f is the given reflex face complexity for f.

Although the modified network H described above is nonplanar for $k \geq 1$, it has linear size in the number of vertices n in G, and has no cost associated with the edges. In the preliminary version of this paper [1], we used the algorithm of Goldberg and Rao [13] to compute a maximum flow for H in $O(n^{1.5} \log n \log k)$ time. An anonymous reviewer pointed out that one can replace each edge (u, v) of H with capacity $c > 1$ by c paths of length two and unit capacity edges, which yields a unit-capacity network with $O(nk)$ vertices and edges, and one can find a maximum flow in such a network in $O(|V|^{1/2}|E|) = O((nk)^{1.5})$ time [10]. Recently, Madry [19] showed that a maximum flow in a network with largest integer capacity U can be computed in $\widetilde{O}(|E|^{10/7}U^{1/7})$ time. Since H has $O(n)$ vertices and edges, and each edge has $O(k)$ capacity upper bound, the running time can be expressed as $\widetilde{O}(n^{10/7}k^{1/7})$.

For the case when $k = 0$, we can find a planar network by deleting the unfilled red vertices, i.e., v'_f, along with the incident edges. Thus the problem reduces to finding a maximum flow in a planar network with multiple sources and sinks, which can be computed in $O(n \log^2 n)$ time [17] since the productions and demands of all the vertices of the network are known.

The following theorem summarizes the main result of this section.

Theorem 2. *Let G be an n-vertex biconnected plane graph with the outer face f_0 and inner faces $f_1, \ldots, f_r$. Given the nonnegative integers $k_0(\geq 4), \ldots, k_r$ with $k = \max_i\{k_i\}$, one can decide in $\widetilde{O}(n^{10/7}k^{1/7})$ time whether G has a strict-orthogonal drawing, where each face f_i has at most k_i reflex corners, and construct such a drawing if it exists.*

4. NP-Hardness for Planar Graphs

In this section we prove that it is NP-complete to decide whether a planar biconnected graph admits a strict-orthogonal drawing with a given reflex face complexity k, even when $k = 4$. Throughout this section we denote this problem by Min-Reflex-Draw.

Garg and Tamassia [12] proved that it is NP-hard to decide whether a maximum-degree-4 planar graph admits a strict-orthogonal drawing. This NP-hardness proof readily implies the NP-hardness of the problem of computing a strict-orthogonal drawing with reflex face complexity k, but this proof does not hold if we restrict k to be a constant. On the other hand, our NP-hardness proof holds when $k = 4$, even when it is known that the input graph has a strict-orthogonal drawing.

We prove the NP-completeness with a reduction from a variation of planar 3-SAT problem (MP3SAT4), which is NP-hard [7]. The input of an MP3SAT4 instance I is a collection C of clauses over a set U of variables such that:

- Each clause contains either two or three variables;
- Each variable appears in at most four clauses, and is negated exactly once;
- Each clause is either positive or negative (i.e., all its variables are either positive or negative);
- The corresponding *SAT-graph* G_I (i.e., the bipartite graph with vertex set $C \cup U$ and edge set $\{(x, y) | x \in C, y \in U, y \in x\}$) admits a planar drawing.

The MP3SAT4 problem asks to decide whether there is a satisfying truth assignment for U satisfying all clauses in C.

Given an instance $I = (U, C)$ of MP3SAT4, where each variable appears in at most four clauses and negated exactly once, we construct a planar graph H so that H has a strict-orthogonal drawing with face complexity 4, if and only if the MP3SAT4 instance is satisfiable.

Every planar graph with n vertices and with maximum degree four admits a planar orthogonal drawing on a grid of size $n \times n$, and such a drawing can be

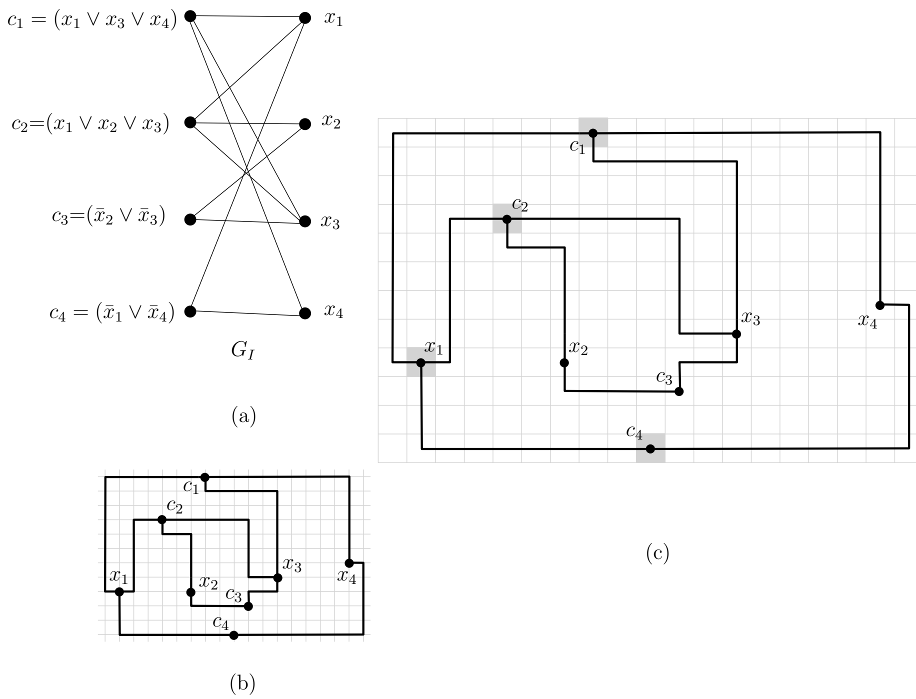


Figure 6: Illustration for (a) G_I, (b) Γ, and (c) Γ'.

computed in linear time [2]. Since G_I is a graph of maximum degree four, we can compute a planar orthogonal drawing Γ of G_I on a polynomial-size grid. We construct H from the drawing Γ. Fig. 6(a) illustrates a SAT-graph G_I and Fig. 6(b) depicts an orthogonal drawing Γ of G_I.

We first scale the drawing Γ by a factor of 4 both horizontally and vertically. Let Γ' be the resulting drawing. Fig. 6(c) depicts a schematic representation of Γ'. Initially, we define H to be the grid graph underlying Γ', where we delete the rows and columns that originally belong to Γ. We now add more vertices and edges to H. For each variable and each clause, we assign a corresponding variable cell and a corresponding clause cell in H. For example, the cells corresponding to the variable x_1 and clauses c_1, c_2, c_4 are shown in gray. Since each variable x appears negated exactly once, one side of the variable cell is intersected by an edge that connects x to a negative clause. We refer to this side as the *heavy side* of the variable cell. For example, in Fig. 7(a), the bottom side of the variable cell for x_1 is a heavy side. In each variable cell, we create a variable-staircase structure of length three, (see Fig. 7(b)), such that the base of the staircase is adjacent to the heavy side of the cell. Note that this staircase contributes to four reflex corners in the variable cell, which can be transferred to the other cell adjacent to the heavy side by flipping the staircase. For each edge e connecting a variable to a clause, we first find the sequence of cells intersected by the drawing of e, and then add a staircase of length two and a 5×5 grid

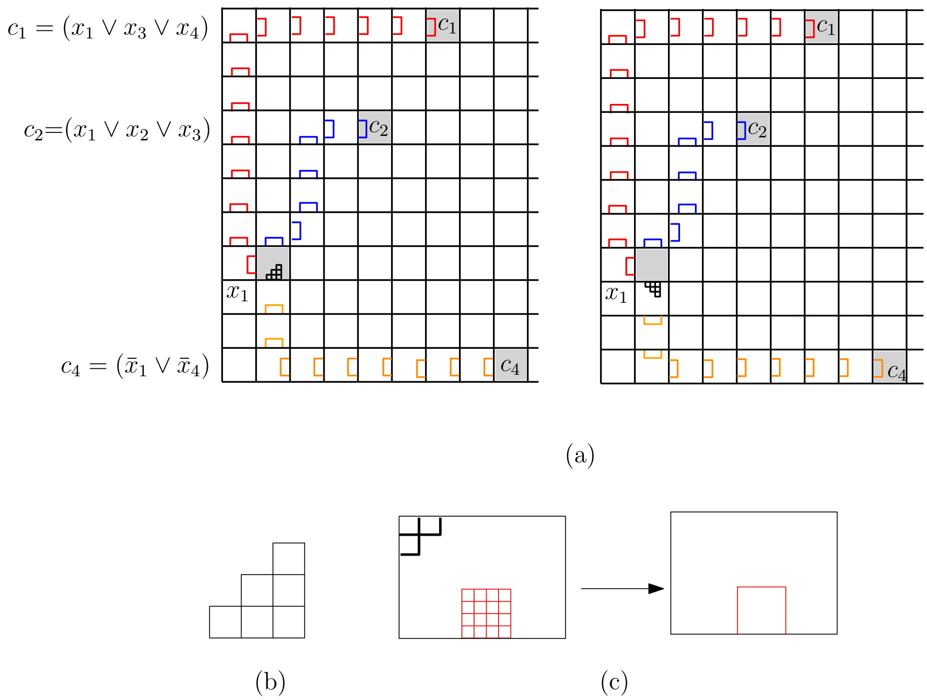


Figure 7: (a) Illustration of the reduction, where the variable and clause cells are shaded. (b) A staircase of length three. (c) A cell with a staircase of length two and a grid structure.

structure (see Fig. 7(c)) to each of these cells, as described below.

The staircase is added at a corner of the cell that cannot be flipped and contributes to two reflex corners of the cell (these staircases are not shown in the schematic representations of Figs. 7–8 in order to preserve the clarity of the drawing). The grid structure is added to those sides that are intersected by the drawing of e, but not a heavy side. Consequently, if a grid structure belong to some cell w and attached to some side s of w, then it contributes to two reflex corners of w, which can be transferred to the other cell adjacent to s by flipping. Since $k = 4$, none of the cells on the path from the variable to the clause cell can contain more than one grid structure. The grid structures are added exploiting this constraint along the variable to clause path, so that if the clause is positive (resp. negative), then the placement of the variable-staircase inside (resp., outside) the variable cell eventually forces a grid structure to fall into the corresponding clause cell, e.g., see Fig. 7(a).

Finally, for each clause c, we add a staircase of length $(6 - 2|c|)$ at the corner of its clause cell, where $|c|$ is the number of variables in c. Such a clause-staircase ensures that at least one of the grid structures incident to the clause cell must lie outside of the clause cell (these staircases are not shown in the schematic representations of Figs. 7–8).

Let the resulting drawing be Γ', as illustrated in Fig. 8(a). It is straightforward to carry out the above construction in polynomial time, and one can observe that any strict-orthogonal drawing must respect the axis-alignments of the edges of the underlying graph (up to rotation or reflection).

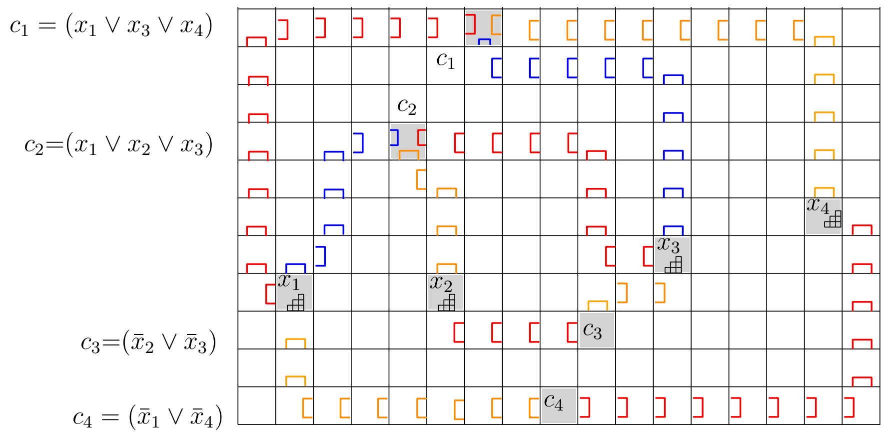
Theorem 3. *It is NP-complete to decide if a planar graph admits a strict-orthogonal drawing with reflex face complexity* 4*.*

Proof: By [23], for any orthogonal drawing of H, one can compute a topologically equivalent drawing where the vertices and bends are on integer coordinates. Therefore given a drawing Γ_H of H (on integer coordinates), it is straightforward to decide in polynomial time if Γ is a strict-orthogonal drawing with reflex face complexity 4. Thus Min-Reflex-Draw is in NP. We now reduce the MP3SAT4 problem to Min-Reflex-Draw.

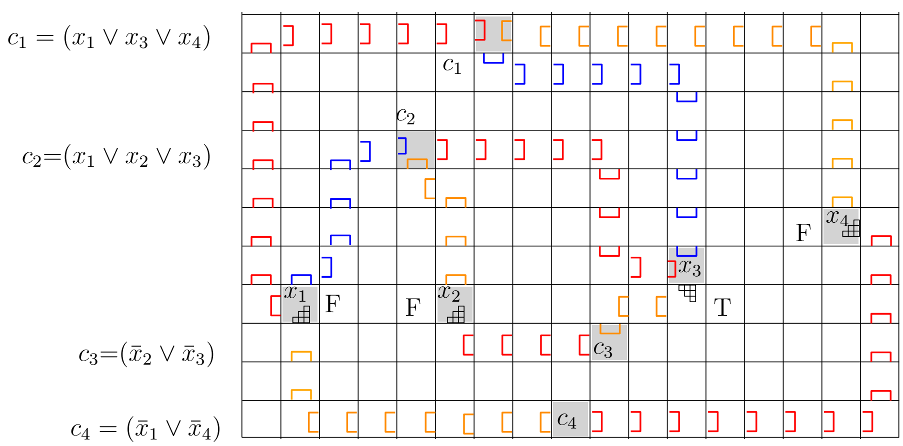
Let $I = (U, C)$ be an instance of MP3SAT4, and let H be the corresponding planar graph. We now prove that H admits a strict-orthogonal drawing with face complexity 4, if and only if the MP3SAT4 instance is satisfiable.

Given a drawing of H with reflex face complexity 4, we assign the truth value of a variable depending on whether the corresponding variable staircase is inside or outside of the variable cell; see Fig. 8(b). By construction, no clause cell can have all its adjacent grid structures inside it, otherwise it would have at least $(6 - 2|c|) + 2|c| > 4$ reflex corners. Consequently, every clause cell must have one of its grid-structures M outside of the clause cell. Recall that any variable cell that receives a variable staircase obtains at least 4 reflex corners, and hence cannot have any grid structure inside it. Therefore, the grid structure M will force the corresponding variable staircase to lie outside or inside of its variable-cell depending on whether the clause is positive or negative. We assign the outside and inside configurations the values true and false, respectively, which implies that each clause must be satisfied.

On the other hand, given a satisfying truth assignment for I, we orient the variable-staircases inside or outside depending on whether it is false or true. The placement for the grid structures is then straightforward, which is guided by the restrictions on the variable to clause paths. Therefore, to verify that the reflex face complexity is bounded by 4, we only need to examine the clause cells. In the following we show that each clause cell with more than 4 reflex corners can be locally modified so that the modified cell contains at most 4 reflex corners, without inducing more than 4 reflex corners in any other cell. Let c be a clause that contains all its incident grid-structures inside the cell yielding $(6 - 2|c|) + 2|c| > 4$ reflex face complexity. Without loss of generality, assume that the clause is positive. Since c is satisfied, at least one of its variable-staircase must lie outside of its variable cell. We now can choose this variable-to-clause path to flip a grid structure out of the clause cell of c.
□



(a)



(b)

Figure 8: (a) Illustration of the reduction, where the variable and clause cells are shaded. (b) computing truth assignment: $x_1 = x_2 = x_4$ =false, x_3=true.

5. Conclusion

Motivated by the problem of rectilinear schematization of maps, we consider two natural variants: one when we are given the same "allowance" of corners for each region, and the another, when each region has its own number of corners. We described two polynomial-time algorithms to compute a solution (or report that one does not exist). If the largest number of allowed corners over all the regions is k, then our matching-based algorithm takes $\widetilde{O}((nk)^{10/7})$-time, and the flow-based algorithm takes $\widetilde{O}(n^{10/7}k^{1/7})$-time.

One potential direction for speeding up our matching-based approach could be based on the concept of a b-matching. Given a graph G in which each vertex v has a degree bound $b(v)$, the b-matching problem asks to to find a maximum cardinality set of edges $M \subseteq E$, such that v is incident to at most $b(v)$ edges of M. Madry [19] proved that a maximum bipartite b-matching with maximum degree bound β can be computed in $\widetilde{O}(|E|^{10/7}\beta^{1/7})$ time. Therefore, a promising strategy would be to reduce the problem of computing a perfect matching in $B(G)$ (see Section 2.2) to the problem of computing a maximum bipartite b-matching in some linear size graph, where the maximum degree bound for each vertex is k.

We also showed that in the variable-embedding setting the problem of deciding whether a biconnected planar graph admits a strict-orthogonal drawing with a given reflex face complexity 4 is NP-complete. Therefore, it would be worthwhile to consider the complexity of the problem for specific values of k, where $k < 4$.

Acknowledgments

We thank the anonymous reviewers for pointing out how network-flow formulations from earlier work can be modified to compute orthogonal drawings with bounded reflex face complexities, and for the suggestions on improving the NP-hardness result.

References

[1] M. J. Alam, S. G. Kobourov, and D. Mondal. Orthogonal layout with optimal face complexity. In *Proceedings of the Forty-First International Conference on Current Trends in Theory and Practice of Computer Science (SOFSEM)*, LNCS, pages 121–133. Springer, 2016.

[2] T. C. Biedl and G. Kant. A better heuristic for orthogonal graph drawings. *Computational Geometry*, 9(3):159–180, 1998.

[3] T. Bläsius, M. Krug, I. Rutter, and D. Wagner. Orthogonal graph drawing with flexibility constraints. *Algorithmica*, 68(4):859–885, 2014.

[4] T. Bläsius, I. Rutter, and D. Wagner. Optimal orthogonal graph drawing with convex bend costs. *ACM Trans. Algorithms*, 12(3):33:1–33:32, 2016.

[5] M. Chimani, C. Gutwenger, M. Jünger, G. Klau, K. Klein, and P. Mutzel. The open graph drawing framework. In *Handbook of Graph Drawing and Visualization*, pages 543–571. 2013.

[6] S. Cornelsen and A. Karrenbauer. Acclerated bend minimization. *Journal of Graph Algorithms and Applications*, 16(3):635–650, 2012.

[7] A. Darmann, J. Döcker, and B. Dorn. On planar variants of the monotone satisfiability problem with bounded variable appearances. *CoRR*, abs/1604.05588, 2016. http://arxiv.org/abs/1604.05588.

[8] G. Di Battista, P. Eades, R. Tamassia, and I. G. Tollis. *Graph Drawing: Algorithms for the Visualization of Graphs.* The MIT Press, 3rd edition, 2009.

[9] J. Ellson, E. R. Gansner, E. Koutsofios, S. C. North, and G. Woodhull. Graphviz - open source graph drawing tools. In *Proceedings of the 9th International Symposium on Graph Drawing (GD)*, volume 2265 of *LNCS*, pages 483–484. Springer, 2001.

[10] S. Even and R. E. Tarjan. Network flow and testing graph connectivity. *SIAM Journal on Computing*, 4(4):507–518, 1975.

[11] U. Fößmeier and M. Kaufmann. Drawing high degree graphs with low bend numbers. In *Symposium on Graph Drawing (GD)*, volume 1027 of *LNCS*, pages 254–266. Springer, 1995.

[12] A. Garg and R. Tamassia. On the computational complexity of upward and rectilinear planarity testing. *SIAM Journal on Computing*, 31(2):601–625, 2001.

[13] A. V. Goldberg and S. Rao. Beyond the flow decomposition barrier. *Journal of the ACM*, 45(5):783–797, 1998.

[14] J. E. Hopcroft and R. M. Karp. An $n^{5/2}$ algorithm for maximum matchings in bipartite graphs. *SIAM Journal on Computing*, 2(4):225–231, 1973.

[15] M. Kaufmann and D. Wagner. *Drawing Graphs: Methods and Models*, volume 2025 of *LNCS*. Springer-Verlag, London, UK, 2001.

[16] S. Kieffer, T. Dwyer, K. Marriott, and M. Wybrow. HOLA: human-like orthogonal network layout. *IEEE Transactions on Visualization and Computer Graphics*, 22(1):349–358, 2016.

[17] P. N. Klein, S. Mozes, and O. Weimann. Shortest paths in directed planar graphs with negative lengths: A linear-space $O(n \log^2 n)$-time algorithm. *ACM Transactions on Algorithms*, 6(2):236–245, 2010.

[18] C. E. Leiserson. Area-efficient graph layouts (for VLSI). In *Symposium on Foundations of Computer Science (FOCS)*, pages 270–281, 1980.

[19] A. Madry. Computing maximum flow with augmenting electrical flows. In *Proceedings of the IEEE 57th Annual Symposium on Foundations of Computer Science (FOCS)*, pages 593–602. IEEE Computer Society, 2016.

[20] K. Miura, H. Haga, and T. Nishizeki. Inner rectangular drawings of plane graphs. *International Journal of Computational Geometry and Applications*, 16(2–3):249–270, 2006.

[21] T. Nishizeki and M. S. Rahman. *Planar Graph Drawing*. World Scientific, Singapore, 2004.

[22] M. S. Rahman, N. Egi, and T. Nishizeki. No-bend orthogonal drawings of subdivisions of planar triconnected cubic graphs. *IEICE Transactions*, 88-D(1):23–30, 2005.

[23] R. Tamassia. On embedding a graph in the grid with the minimum number of bends. *SIAM Journal on Computing*, 16(3):421–444, 1987.

[24] W. Tobler. Thirty five years of computer cartograms. *Annals of Association of American Geographers*, 94:58–73, 2004.

[25] L. G. Valiant. Universality considerations in VLSI circuits. *IEEE Transaction on Computers*, 30(2):135–140, 1981.

[26] R. Wiese, M. Eiglsperger, and M. Kaufmann. yfiles: Visualization and automatic layout of graphs. In *Proceedings of the 9th International Symposium on Graph Drawing*, volume 2265 of *LNCS*, pages 453–454. Springer, 2001.