Aspect Ratio Universal Rectangular Layouts

Stefan Felsner ✉
Institut für Mathematik, Technische Universität Berlin, Berlin, Germany

Andrew Nathenson ✉
University of California, San Diego, CA, USA

Csaba D. Tóth ✉
Department of Mathematics, California State University Northridge, Los Angeles, CA; and
Department of Computer Science, Tufts University, Medford, MA, USA

─── **Abstract** ───

A *generic rectangular layout* (for short, *layout*) is a subdivision of an axis-aligned rectangle into axis-aligned rectangles, no four of which have a point in common. Such layouts are used in data visualization and in cartography. The contacts between the rectangles represent semantic or geographic relations. A layout is weakly (strongly) *aspect ratio universal* if any assignment of aspect ratios to rectangles can be realized by a weakly (strongly) equivalent layout. We give combinatorial characterizations for weakly and strongly aspect ratio universal layouts. Furthermore, we describe a quadratic-time algorithm that decides whether a given graph is the dual graph of a strongly aspect ratio universal layout, and finds such a layout if one exists.

1 Introduction

A *rectangular layout* (for short, *layout*) is a subdivision of an axis-aligned rectangle into axis-aligned rectangular faces; it is also known as *mosaic floorplan* or *rectangulation*. A layout is *generic* if no four faces have a point in common; see Fig. 1. In this paper all layouts are generic unless stated otherwise. In the *dual graph* $G(\mathcal{L})$ of a layout $\mathcal{L}$, the nodes correspond to rectangular faces, and an edge corresponds to a pair of faces that share a boundary segment of positive length [6, 29, 30].
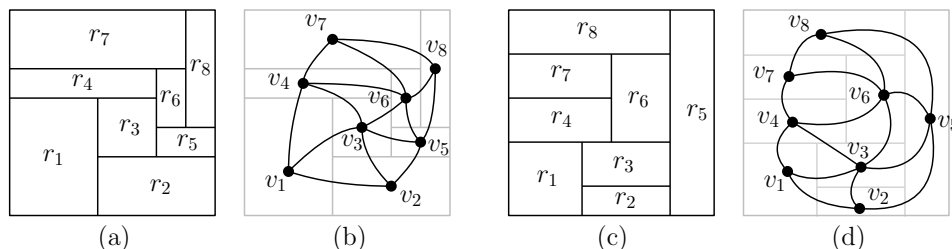
Figure 1 (a–b) A layout and its dual graph. (c–d) A sliceable layout and its dual graph. The two layouts are neither strongly nor weakly equivalent, but their dual graphs are isomorphic.

Two layouts are *strongly equivalent* if they have isomorphic dual graphs, and the corresponding line segments between adjacent faces have the same orientation (horizontal or vertical). Two layouts are *weakly equivalent* if there are bijections between their maximal horizontal line segments as well as between their maximal vertical line segments, and the contact graphs of all segments are isomorphic plane graphs. Strong equivalence implies weak equivalence [9], but weak equivalence does not imply strong equivalence; see Fig. 2 for examples. The closures of weak (resp., strong) equivalence classes under the Hausdorff distance[1] extend to nongeneric layouts, and a nongeneric layout may belong to the closures of multiple equivalence classes.
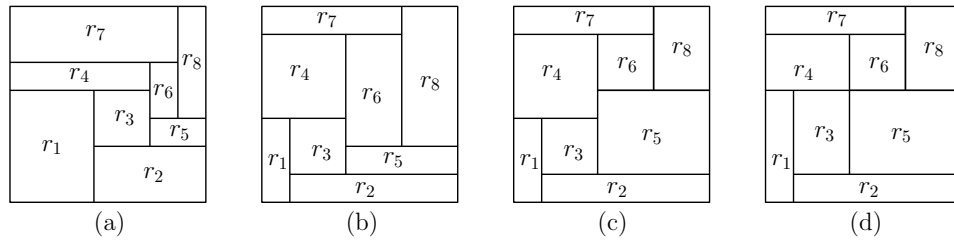
Figure 2 (a–c) Three generic layouts that are weakly equivalent. The layouts in (a) and (b) are strongly equivalent, but not strongly equivalent to the layout in (c), as the adjacencies between the pairs of rectangles $\{r_3, r_6\}$ and $\{r_4, r_5\}$ are different. (d) A nongeneric layout that is in the closure of the strong equivalence classes of the layouts in (a–b) and (c).

Rectangular layouts have been studied for more than 40 years, originally motivated by VLSI design [22, 24, 36] and cartography [27], and more recently by data visualization [18]. The weak equivalence classes of layouts are in bijection with Baxter permutations [1, 28, 37].

A graph is called a *proper graph* if it is the dual graph of a generic layout. Every proper graph is a near-triangulation (a plane graph where every bounded face is a triangle, but the outer face need not be a triangle). But not every near-triangulation is a proper graph [29, 30]. Ungar [34] gave a combinatorial characterization of proper graphs (see also [17, 32]): A near-triangulation is a proper graph if and only if it can be augmented by four vertices into a near-triangulation in which the outer face is a quadrilateral formed by the new vertices, and there are no separating triangles. Using this characterization, proper graphs can be recognized in linear time [12, 23, 25, 26].

In data visualization and cartography [18, 27], the rectangular faces correspond to entities (e.g., countries or geographic regions); adjacency between rectangles represents semantic or geographic relations, and the "shape" (e.g., area, width, height, aspect ratio, in-radius, etc.) of a rectangular face represents data associated with the entity. It is often desirable to use equivalent layouts to realize different statistics associated with the same entities. A generic layout $\mathcal{L}$ is *weakly (strongly) area universal* if any area assignment to the rectangles can be realized by a layout weakly (strongly) equivalent to $\mathcal{L}$. Wimer et al. [36] showed that every generic layout is weakly area universal (see also [9, Thm. 3]). Eppstein et al. [6] proved that a layout is strongly area universal if and only if it is one-sided (defined below). However, no polynomial-time algorithm is known for testing whether a given graph G is the dual graph of some area-universal layout.

---

[1] The distance between layouts $\mathcal{L}_1$ and $\mathcal{L}_2$ is the Hausdorff distance between the sets $S(\mathcal{L}_1)$ and $S(\mathcal{L}_2)$, where $S(\mathcal{L}_i)$ is the union of all segments in $\mathcal{L}_i$ for $i \in \{1, 2\}$.

Aspect ratio universal layouts. The *aspect ratio* of an axis-aligned rectangle $r = [a, b] \times [c, d]$ is $\text{height}(r)/\text{width}(r) = (d - c)/(b - a)$. In some applications, the aspect ratios (rather than the areas) of the rectangles are specified. For example, in word clouds adapted to multiple languages, the aspect ratio of (the bounding box of) each word depends on the particular language. A generic layout $\mathcal{L}$ is *weakly* (*strongly*) *aspect ratio universal* (*ARU*, for short) if any assignment of aspect ratios to the rectangles can be realized by a layout that is weakly (strongly) equivalent to $\mathcal{L}$.

In this paper, we give a combinatorial characterization for weakly and strongly ARU layouts, and design an algorithm to recognize strongly ARU layouts. Our results are stated in Section 1.2.

1.1 Background and terminology

Before we can state our results, we review some additional key definitions for layouts. A *rectilinear graph* is a plane straight-line graph in which every edge is horizontal or vertical. A *rectangular layout* (for short, *layout*) is a rectilinear graph in which the boundary of every face (including the outer face) is a rectangle; it is *generic* if the maximum vertex degree is at most three. A *sublayout* of a layout $\mathcal{L}$ is a subgraph of $\mathcal{L}$ which is a layout. A sublayout $\mathcal{L}'$ of $\mathcal{L}$ is *trivial* if $\mathcal{L}' = \mathcal{L}$ or $\mathcal{L}'$ is the boundary of a single face of $\mathcal{L}$. A layout $\mathcal{L}$ is *irreducible* if it does not have any nontrivial sublayout. A *rectangular arrangement* is a 2-connected plane straight-line graph in which every bounded face is a rectangle (however, the outer face need not be a rectangle); see Fig. 3 for examples.

(a)   (b)   (c)   (d)
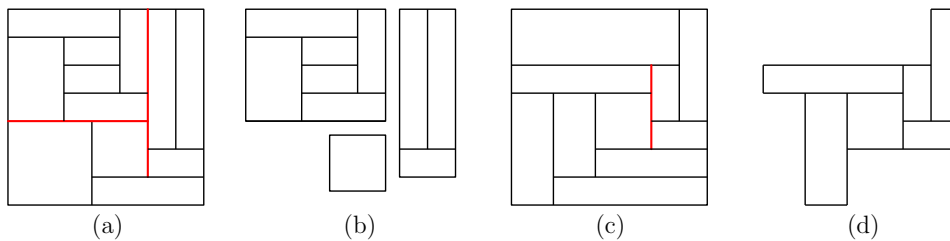
■ **Figure 3** (a) A layout $\mathcal{L}$, the red maximal segments are not one-sided. (b) Three sublayouts of $\mathcal{L}$, one of which is trivial. (c) An irreducible layout. (d) A rectangular arrangement.

One-Sided layouts. A *segment* of a layout $\mathcal{L}$ is a path of collinear inner edges of $\mathcal{L}$. A segment of $\mathcal{L}$ that is not contained in any other segment is *maximal*. A layout $\mathcal{L}$ is *one-sided* if every maximal segment is a side of at least one rectangular face of $\mathcal{L}$; that is, for every maximal segment s, all other segments with an endpoint in the interior of s lie in the same halfplane bounded by s. For example, the layouts in Fig. 3a and 3c are not one-sided (due to the red maximal segments).

Sliceable layouts. A *slice* in a layout $\mathcal{L}$ is a maximal segment with both endpoints incident to the outer face, and subdivides $\mathcal{L}$ into two sublayouts. A *sliceable layout* (also known as *slicing floorplan* or *guillotine rectangulation*) is one that can be decomposed into trivial layouts through recursive subdivision with slices; see Fig 1(c) for an example. The recursive subdivision of a layout $\mathcal{L}$ can be represented by a *binary space partition tree* (*BSP-tree*), which is a binary tree where each vertex is associated with a sublayout, which is $\mathcal{L}$ if the vertex is the root and a rectangular face of $\mathcal{L}$ if the vertex is a leaf [5]. For a nonleaf vertex,

the tree additionally stores a slice; and two sublayouts on each side of the slice are associated with the two children. The *root slice* is between two opposite sides of the bounding box.

The number of (strong equivalence classes of) sliceable layouts with n rectangular faces is known to be the nth Schröder number [37]. One-sided sliceable layouts are in bijection with certain pattern-avoiding permutations, closed formulas for their number have been given by Asinowski and Mansour [2]; see also [21] and OEIS A078482 in the on-line encyclopedia of integer sequences for further references.

A *windmill* in a layout is a set of four pairwise noncrossing maximal segments, called *arms*, which contain the sides of a central rectangle and each arm has an endpoint on the interior of another (e.g., the maximal segments around the rectangular face r_3 or r_6 in Fig. 4a). A windmill is either *clockwise* or *counterclockwise*, depending on the orientation of the boundary of the central rectangle when we orient each arm from the central rectangle to the other endpoint (see Figs. 6e–6f). It is well known that a layout is sliceable if and only if it does not contain a windmill [1].

Transversal structure. The dual graph $G(\mathcal{L})$ of a layout $\mathcal{L}$ encodes adjacencies between faces, but does not specify the relative positions between faces (above-below or left-right). The transversal structure (also known as *regular edge-labelling*) was introduced by He [13, 15] for the efficient recognition of proper graphs, and later used extensively for counting and enumerating (equivalence classes of) layouts. Fusy [10, Section 2] distinguishes between several equivalent variants of the transversal structure, we refer to the variant called *transversal edge-partition* by Fusy [10].

We first define the *extended dual graph* $G^*(\mathcal{L})$ of a layout $\mathcal{L}$, as the contact graph of the rectangular faces *and* the four edges of the bounding box of $\mathcal{L}$; it is a triangulation in an outer 4-cycle without separating triangles; see Fig. 4.
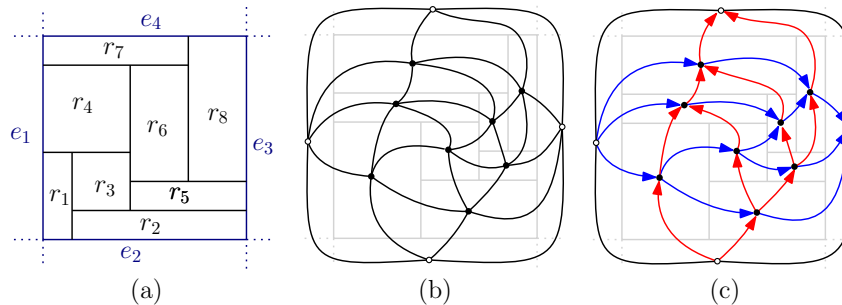
Figure 4 (a) A layout $\mathcal{L}$ bounded by edges $e_1, \ldots, e_4$. (b) Extended dual graph $G^*(\mathcal{L})$ with an outer 4-cycle $(e_1, \ldots, e_4)$. (c) A transversal structure.

The *transversal structure* of a layout $\mathcal{L}$ comprises $G^*(\mathcal{L})$ and a bicoloring of the inner edges of $G^*(\mathcal{L})$, where red (resp., blue) edges correspond to above-below (resp., left-right) relation between two objects in contact. An (abstract) *transversal structure* is defined as a graph G^*, which is a triangulation of an outer 4-cycle (S, W, N, E) that has no separating triangles, together with a bicoloring of the inner edges of G^* such that all the inner edges incident to S, W, N, and E are red, blue, red, and blue, respectively; and at each inner vertex the counterclockwise rotation of incident edges consists of four nonempty blocks of red, blue, red, and blue edges; see Fig. 4c. The transversal structure determines a unique orientation of the edges [10, Proposition 2], where red (resp., blue) edges are directed bottom-up (resp., left-to-right), as indicated in Fig. 4c. It is known that transversal structures are in bijection

with the strong equivalence classes of generic layouts [8, 10, 15].
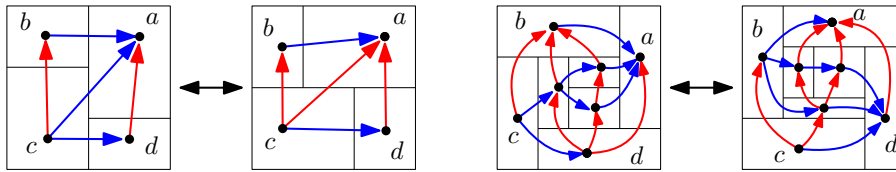
Figure 5 A flip of an empty (left) and a nonempty (right) alternating cycle.

Flips and alternating 4-cycles. A sequence of flip operations can transform any transversal structure with n inner vertices into any other [7, 10]. Each *flip* considers an *alternating 4-cycle C*, which comprises red and blue edges alternatingly, and changes the color of every edge in the interior of C; see Fig. 5. If, in particular, there is no vertex in the interior of C, then the flip changes the color of the inner diagonal of C. Furthermore, every flip operation yields a valid transversal structure on $G^*(\mathcal{L})$, hence a new generic layout $\mathcal{L}'$ that is strongly nonequivalent to $\mathcal{L}$. We can now establish a relation between geometric and combinatorial properties.

▶ **Lemma 1.** *A layout $\mathcal{L}$ is one-sided and sliceable if and only if $G^*(\mathcal{L})$ admits a unique transversal structure.*

Proof. Assume that $\mathcal{L}$ is a layout where $G^*(\mathcal{L})$ admits two or more transversal structures. Consider a transversal structure of $G^*(\mathcal{L})$. Since any two transversal structures are connected by a sequence of flips, there exists an alternating 4-cycle. Any alternating 4-cycle with no interior vertex corresponds to a segment in $\mathcal{L}$ that is not one-sided. Any alternating 4-cycle with interior vertices corresponds to a windmill in $\mathcal{L}$. Consequently, $\mathcal{L}$ is not one-sided or not sliceable.

Conversely, if $\mathcal{L}$ is not one-sided (resp., sliceable), then the transversal structure of $G^*(\mathcal{L})$ contains an alternating 4-cycle with no interior vertex (resp., with interior vertices). Consequently, we can perform a flip operation, and obtain another transversal structure for $G^*(\mathcal{L})$. ◀

1.2 Our results

We characterize strongly and weakly aspect ratio universal layouts.

▶ **Theorem 2.** *A generic layout is weakly aspect ratio universal if and only if it is sliceable.*

▶ **Theorem 3.** *For a generic layout $\mathcal{L}$, the following properties are equivalent:*
(i) $\mathcal{L}$ is strongly aspect ratio universal;
(ii) $\mathcal{L}$ is one-sided and sliceable;
(iii) the extended dual graph $G^*(\mathcal{L})$ of $\mathcal{L}$ admits a unique transversal structure.

It is not difficult to show that one-sided sliceable layouts are strongly aspect ratio universal; and admit a unique transversal structure. Proving the converses, however, is more involved.

Algorithmic results. In some applications, the rectangular layout is not specified, and we are only given the dual graph of a layout (i.e., a proper graph). This raises the following problem: Given a (proper) graph G, find a strongly (resp., weakly) ARU layout $\mathcal{L}$ such that G is isomorphic to the dual graph of $\mathcal{L}$ (i.e., $G \simeq G(\mathcal{L})$) or report that no such layout exists.

Using structural properties of one-sided sliceable layouts that we develop here, we present an algorithm for recognizing the dual graphs of strongly ARU layouts.

▶ **Theorem 4.** *We can decide in $O(n^2)$ time whether a given graph G with n vertices is the dual of a one-sided sliceable layout.*

   Currently, no polynomial-time algorithm is known for recognizing dual graphs of sliceable layouts [4, 19, 38] (which are weakly ARU layouts by Theorem 2); or one-sided layouts [6]. Previously, Thomassen [32] gave a linear-time algorithm to recognize proper graphs if the nodes corresponding to corner rectangles are specified, using combinatorial characterizations of layouts [34]. Kant and He [13, 15] described a linear-time algorithm to test whether a given graph G^* is the extended dual graph of a layout, using transversal structures. Later, Rahman et al. [12, 23, 25, 26] showed that proper graphs can be recognized in linear time (without specifying the corners). However, a proper graph may have exponentially many (strongly or weakly) nonequivalent realizations, and none of the prior algorithms guarantees to find a one-sided sliceable realization if one exists.

Organization.   We characterize strongly and weakly ARU layouts and prove Theorems 2–3 in Section 2. We establish structural properties of the dual graphs of one-sided sliceable layouts in Subsection 3.1, and use them in the analysis of a quadratic-time algorithm that recognizes such graphs in Subsection 3.2. We conclude in Section 4 with open problems and one more structural property: We show that the dual graph of every one-sided sliceable layout has a vertex cut of size at most 3 (Proposition 23).

2   Aspect ratio universality

An *aspect ratio assignment* to a layout $\mathcal{L}$ is a function α that maps a positive real to each rectanglular face in $\mathcal{L}$. A strong (resp., weak) *realization* of an aspect ratio assignment α to $\mathcal{L}$ is a layout $\mathcal{L}'$ that is strongly (resp., weakly) equivalent to $\mathcal{L}$ with the required aspect ratios. If such a realization exists, then the aspect ratio assignment α is strongly (resp., weakly) *realizable*. A layout is strongly (resp., weakly) *aspect ratio universal* (*ARU*) if every aspect ratio assignment is strongly (resp., weakly) realizable. In this section, we characterize strongly and weakly ARU layouts (Theorems 2 and 3). We start with two easy observations about one-sided and sliceable layouts.

▶ **Lemma 5.** *Let $\mathcal{L}$ be a layout with an aspect ratio assignment α. Then every strong realization is a weak realization. Furthermore, if $\mathcal{L}$ is one-sided, then every weak realization is a strong realization.*

Proof. The first statement follows from the fact that strong equivalence implies weak equivalence. For the second statement, note that in a one-sided layout, each maximal segment s is a side of a rectangular face, and so s is on the boundary between the same pairs of rectangular faces in every weak realization of α. Consequently, all weak realizations of α generate the same dual graph, hence they are all strongly equivalent to $\mathcal{L}$, and so they are strong realizations of α.   ◀

▶ **Lemma 6.** *Every aspect ratio assignment for a sliceable layout has a unique weak realization up to scaling and translation. Furthermore, for every $\varrho > 0$ there exists a weakly realizable aspect ratio assignment α such that the bounding box of the unique realization of α has aspect ratio ϱ.*

Proof. To prove the first claim, let α be an aspect ratio assignment to a sliceable layout $\mathcal{L}$. Note that the restriction of α to a sublayout $\mathcal{L}'$ of $\mathcal{L}$ is an aspect ratio assignment for $\mathcal{L}'$. We proceed by induction on k, the height of the BSP-tree representing $\mathcal{L}$.

In the basis step, a layout of height 0 comprises a layout with a single rectangular face, which is uniquely determined by its aspect ratio assignment up to scaling and translation. For the induction step, assume that every sublayout at height k in the BSP-tree admits a unique weak realization in which all rectangular faces (at the leaves of the BSP-tree) have the required aspect ratios. A sublayout $\mathcal{L}_0$ at height $k+1$ of the BSP-tree is composed of two sublayouts at height k, say $\mathcal{L}_1$ and $\mathcal{L}_2$, that share an edge. Given a weak realization of $\mathcal{L}_1$, there is a unique scaling and translation that attaches $\mathcal{L}_2$ to $\mathcal{L}_1$, and identifies their matching boundary segments. This yields a unique weak realization of the sublayout $\mathcal{L}$ at level $k+1$, up to scaling and translation.

The second claim follows trivially: Start with a bounding box of aspect ratio ϱ, subdivide it recursively into a layout equivalent to $\mathcal{L}$ (but otherwise arbitrarily), and define an aspect ratio assignment using the aspect ratios of the resulting leaf rectangles. ◀

By the definition of weak ARU, Lemma 6 readily implies the following.

▶ **Corollary 7.** *If $\mathcal{L}$ is sliceable, then it is weakly ARU.*

The combination of Lemmata 5 and 6 yields the following for one-sided sliceable layouts.

▶ **Corollary 8.** *If $\mathcal{L}$ is one-sided and sliceable, then it is strongly ARU.*

2.1 Sliceable and One-Sided Layouts

Next we show that any sliceable layout that is strongly ARU must be one-sided. We present two types of elementary layouts that are not strongly ARU, and then show that all other layouts that are not one-sided or not sliceable can be reduced to these prototypes. A *brick layout* is a sliceable layout whose BSP-tree is a complete binary tree with four leaves, and the slices at consecutive levels are orthogonal; see Figs. 6a–6d. A *windmill layout* is generated by the four arms of a (clockwise or counterclockwise) windmill, where each arm extends to the outer boundary; see Figs. 6e–6f.

(a)　　　**(b)**　　　**(c)**　　　**(d)**　　　**(e)**　　　**(f)**
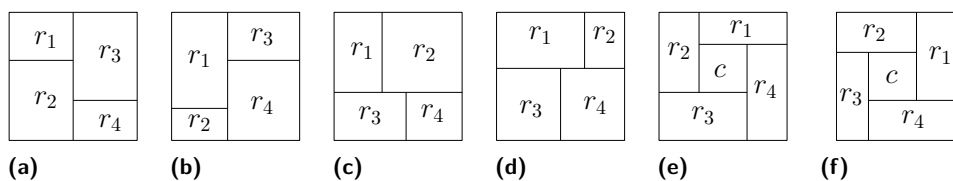
Figure 6 Layouts that are not aspect ratio universal: (a)–(d) brick layouts are sliceable but not one-sided; (e)–(f) windmill layouts are one-sided but not sliceable.

▶ **Lemma 9.** *The brick layouts in Figs. 6a–6d are not strongly ARU; the windmill layouts in Figs. 6e–6f are neither strongly nor weakly ARU.*

Proof. Suppose, for a contradiction, that a brick layout is strongly ARU. We may assume, by symmetry, that the brick layout $\mathcal{L}_0$ in Fig. 6a is strongly ARU. Then there exists a strongly equivalent layout $\mathcal{L}$ for the aspect ratio assignment $\alpha(r_2) = \alpha(r_3) = 1$ and $\alpha(r_1) = \alpha(r_4) = 2$. Since the same horizontal slice is a side of both r_1 and r_2, then width$(r_1) =$ width(r_2). Combined with $\alpha(r_1) = 2\alpha(r_2)$, this yields height$(r_1) = 2\,$height$(r_2)$, and so the left horizontal
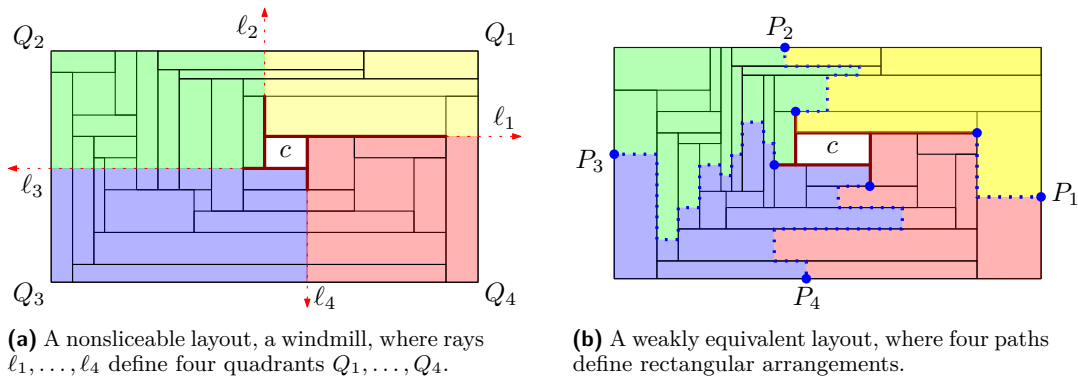
slice is below the barycenter of $r_1 \cup r_2$. Similarly, $\text{width}(r_3) = \text{width}(r_4)$ and $\alpha(r_4) = 2\alpha(r_3)$ imply that the right horizontal slice is above the barycenter of $r_3 \cup r_4$. Since barycenter of $r_1 \cup r_2$ and $r_3 \cup r_4$ have the same y-coordinate, then r_1 and r_4 are in contact in the realization $\mathcal{L}$, which is not strongly equivalent to $\mathcal{L}_0$: a contradiction.

Suppose that a windmill layout is weakly ARU. We may assume, by symmetry, that the windmill layout $\mathcal{L}_1$ in Fig. 6e is weakly ARU. Then there exists a weakly equivalent layout $\mathcal{L}$ for the aspect ratio assignment $\alpha(c) = \alpha(r_1) = \alpha(r_2) = \alpha(r_3) = \alpha(r_4) = 1$. In particular, $r_1, \ldots, r_4$ are squares; denote their side lengths by s_i, for $i = 1, \ldots, 4$. Note that one side of r_i strictly contains a side of r_{i-1} for $i = 1, \ldots, 4$ (with arithmetic modulo 4). Consequently, $s_1 < s_2 < s_3 < s_4 < s_1$, which is a contradiction. ◄

▶ **Lemma 10.** *If a layout is sliceable but not one-sided, then it is not strongly ARU.*

Proof. Let $\mathcal{L}$ be a sliceable but not one-sided layout. It suffices to show that any of its sublayouts are not strongly ARU, because any nonrealizable aspect ratio assignment for a sublayout can be expanded arbitrarily to an aspect ratio assignment for the entire layout.

We claim that $\mathcal{L}$ contains a sublayout strongly equivalent to a brick layout in Figs. 6a–6d. As $\mathcal{L}$ is not one-sided, it contains a maximal segment ℓ which is not the side of any rectangular face. We may assume, without loss of generality, that ℓ is vertical. Since $\mathcal{L}$ is sliceable, every maximal segment is a slice of a sublayout that subdivides it into two smaller sublayouts. However, ℓ is not the side of any rectangular face, consequently the two smaller sublayouts on the left and right of ℓ must be subdivided horizontally in the recursion. Let ℓ_{left} and ℓ_{right} be the first maximal horizontal segments on the left and right of ℓ, respectively. Assume that they each subdivide a sublayout adjacent to ℓ into r_1 and r_2 (on the left) and r_3 and r_4 (on the right). The bounding boxes of the sublayouts $r_1, \ldots, r_4$ comprise a layout equivalent to a brick layout in Figs. 6a–6d. By Lemma 9, there exists an aspect ratio assignment to $\mathcal{L}$ not realizable by a strongly equivalent layout. ◄

(a) A nonsliceable layout, a windmill, where rays $\ell_1, \ldots, \ell_4$ define four quadrants $Q_1, \ldots, Q_4$.

(b) A weakly equivalent layout, where four paths define rectangular arrangements.

Figure 7 The rays $\ell_1, \ldots, \ell_4$ deform into monotone paths in a weakly equivalent layout.

As noted above, every nonsliceable layout contains a windmill [1]. In the remainder of this section we strengthen this claim, and show that every nonsliceable layout contains a windmill together with pairwise disjoint x- or y-monotone paths to some points in the interiors of distinct sides of the bounding box. In a nutshell, our proof goes as follows: Consider an arbitrary windmill in a nonslicable layout $\mathcal{L}$. We subdivide the exterior of the windmill into four *quadrants*, by extending the arms of the windmill into rays $\ell_1, \ldots, \ell_4$ to the bounding box; see Fig. 7a. Each rectangular face of $\mathcal{L}$ lies in a quadrant or in the union of two consecutive quadrants. We assign aspect ratios to each rectangular face based on

which quadrant(s) it lies in. If these aspect ratios can be realized by a layout $\mathcal{L}'$ that is weakly equivalent to $\mathcal{L}$, then the rays $\ell_1, \ldots, \ell_4$ will be "deformed" into x- or y-monotone paths that subdivide $\mathcal{L}'$ into a sublayout in the center of the windmill and four rectangular arrangements, each incident to a unique corner of the bounding box; as in Fig. 7b. We assign the aspect ratios for rectangular faces in $\mathcal{L}'$ so that these arrangements can play the same role as the rectangles $r_1, \ldots, r_4$ in a windmill layout in Figs. 6e–6f. We continue with the details.

We clarify what we mean by a "deformation" of a (horizontal) ray ℓ; see Fig. 7.

▶ **Lemma 11.** *Let a ray ℓ be the extension of a horizontal segment in a layout $\mathcal{L}$ such that ℓ does not contain any other segment and it intersects the rectangular faces $r_1, \ldots, r_k$ in this order. Suppose that $\mathcal{L}'$ is weakly equivalent to $\mathcal{L}$, and the corresponding faces $r'_1, \ldots, r'_k$ of $\mathcal{L}'$ are sliced by horizontal segments $s_1, \ldots, s_k$, respectively. Then there exists an x-monotone path comprised of horizontal edges $s_1, \ldots, s_k$, and vertical edges along vertical segments of the layout $\mathcal{L}'$.*

Proof. Assume, without loss of generality, that the ray ℓ points to the right. Since ℓ does not contain any other segment and it intersects the rectangular faces $r_1, \ldots, r_k$ in this order, then r_i and r_{i+1} are on opposite sides of a vertical segment for $i = 1, \ldots, k-1$. As $\mathcal{L}'$ is weakly equivalent to $\mathcal{L}$, then r'_i and r'_{i+1} are on opposite sides of a vertical segment for $i = 1, \ldots, k-1$; see [9]. In particular, the right endpoint of s_i and the left endpoint of s_{i+1} are on the same vertical segment in $\mathcal{L}'$, for all $i = 1, \ldots k-1$. These vertical segments, together with $s_1, \ldots, s_k$ form an x-monotone path, as required.   ◀

The next lemma allows us to estimate the aspect ratio of the bounding box of a rectangular arrangement in terms of the aspect ratios of individual rectangles.

▶ **Lemma 12.** *If every rectangle in a rectangular arrangement has aspect ratio αm, where $\alpha > 0$ and m is the number of rectangles in the arrangement, then the aspect ratio of the bounding box of the arrangement is at least α and at most αm^2.*

Proof. Consider a rectangular arrangement A with m rectangles and a bounding box R. Let w be the maximum width of a rectangle in A. This implies width$(R) \le mw$. Each rectangle of width w in the arrangement has height $\alpha m w$, and so height$(R) \ge \alpha m w$. The aspect ratio of R is height$(R)/$width$(R) \ge (\alpha m w)/(mw) = \alpha$.

Similarly, let h be the maximum height of a rectangle in A. Then height$(R) \le mh$. Any rectangle of height h in the arrangement A has width $\frac{h}{\alpha m}$, and so width$(R) \ge \frac{h}{\alpha m}$. The aspect ratio of R is height$(R)/$width$(R) \le mh/(\frac{h}{\alpha m}) = \alpha m^2$, as claimed.   ◀

We can now complete the characterization of aspect ratio universal layouts.

▶ **Lemma 13.** *If a layout $\mathcal{L}$ is not sliceable, it is not weakly ARU.*

Proof. We proceed by induction on the number n of rectangles in $\mathcal{L}$. In the basis step, we assume that $\mathcal{L}$ is irreducible. In particular, $\mathcal{L}$ contains no slices, as any slice would create two nontrivial sublayouts. Every nonsliceable layout contains a windmill. Consider an arbitrary windmill in $\mathcal{L}$, assume without loss of generality that it is clockwise and let c be its central rectangle (see Fig. 7a). Note that c is a rectangular face in $\mathcal{L}$, since $\mathcal{L}$ is irreducible. Denote by R the bounding box of $\mathcal{L}$. By extending the arms of the windmill into rays, $\ell_1, \ldots, \ell_4$, we subdivide $R \setminus c$ into four *quadrants*, denoted by $Q_1, \ldots, Q_4$ in counterclockwise order starting with the top-right quadrant.

Note that the interior of any rectangular face of $\mathcal{L}$ intersects at most two rays. Indeed, any two points on two different rays, say $p_i \in \ell_i$ and $p_j \in \ell_j$, span a closed axis-parallel rectangle $a(p_i, p_j)$ that intersects (the closure of) the central rectangle c. Now suppose that both p_i and p_j are in the interior of some rectangular face $r \subset R \setminus c$ of $\mathcal{L}$, then $a(p_i, p_j)$ also lies in the interior of r, and so c would intersect the interior of a rectangular face $r \subset R \setminus c$, which is a contradiction. It follows that every face of $\mathcal{L}$ in $R \setminus c$ lies either in one quadrant or in the union of two consecutive quadrants.

We define an aspect ratio assignment α as follows: Let $\alpha(c) = 1$. If $r \subseteq Q_1$ or $r \subseteq Q_3$, let $\alpha(r) = 6n$; and if $r \subseteq Q_2$ or $r \subseteq Q_4$, let $\alpha(r) = (6n^2)^{-1}$. For a rectangle r split by a ray, we set $\alpha(r) = 6n + (6n^2)^{-1}$ if r is split by the horizontal ray ℓ_1 or ℓ_3; and $\alpha(r) = ((6n)^{-1} + (6n^2))^{-1}$ if split by the vertical ray ℓ_2 or ℓ_4.

Suppose that some layout $\mathcal{L}'$ weakly equivalent to $\mathcal{L}$ realizes α. Split every rectangle of aspect ratio $6n + (6n^2)^{-1}$ in $\mathcal{L}'$ horizontally into two rectangles of aspect ratios $6n$ and $(6n^2)^{-1}$. Similarly, split every rectangle of aspect ratio $((6n)^{-1} + (6n^2))^{-1}$ vertically into two rectangles of aspect ratios $6n$ and $(6n^2)^{-1}$; see Fig. 7b. By Lemma 11, there are four x- or y-monotone paths $P_1, \ldots, P_4$ from the four arms of the windwill to four distinct sides of the bounding box that pass through the splitting segments. The paths $P_1, \ldots, P_4$ subdivide the exterior of the windmill into four rectangular arrangements that we denote by $A_1, \ldots, A_4$, and that each contain a unique corner of the bounding box. By construction, every rectangle in A_1 and A_3 has aspect ratio $6n$, and every rectangle in A_2 and A_4 has aspect ratio $(6n^2)^{-1}$.

Let $R_1, \ldots, R_4$ be the bounding boxes of $A_1, \ldots, A_4$, respectively. By Lemma 12, both R_1 and R_3 have aspect ratios at least 6, and both R_2 and R_4 have aspect ratios at most $\frac{1}{6}$. By construction, the rectangular arrangements $A_1, \ldots, A_4$ each contain an arm of the windmill. This implies that $\mathrm{width}(c) < \min\{\mathrm{width}(R_1), \mathrm{width}(R_3)\}$ and $\mathrm{height}(c) < \min\{\mathrm{height}(R_2), \mathrm{height}(R_4)\}$. Consider the rectangular arrangement comprised of A_1, c, and A_3. It contains two opposite corners of R, and so its bounding box is R. Furthermore, $\mathrm{height}(R) \geq \max\{\mathrm{height}(R_1), \mathrm{height}(R_3)\}$, and

$$
\begin{aligned}
\mathrm{width}(R) &\leq \mathrm{width}(R_1) + \mathrm{width}(c) + \mathrm{width}(R_3) \\
&< 3 \max\{\mathrm{width}(R_1), \mathrm{width}(R_3)\} \\
&\leq 3 \max\left\{ \frac{\mathrm{height}(R_1)}{6}, \frac{\mathrm{height}(R_3)}{6} \right\} = \frac{\max\{\mathrm{height}(R_1), \mathrm{height}(R_3)\}}{2},
\end{aligned}
$$

and so the aspect ratio of R is at least 2. Similarly, the bounding box of the rectangular arrangement comprised of A_2, c, and A_3 is also R, and an analogous argument implies that its aspect ratio must be at most $\frac{1}{2}$. We have shown that the aspect ratio of R is at least 2 and at most $\frac{1}{2}$, a contradiction. Thus the aspect ratio assignment α is not realizable, and so $\mathcal{L}$ is not weakly aspect ratio universal.

Induction step. Assume that $\mathcal{L}$ has a nontrivial sublayout. We distinguish between two cases:

Case 1: $\mathcal{L}$ has a nontrivial irreducible sublayout. Let $\mathcal{L}^*$ be a minimum nontrivial irreducible sublayout sublayout of $\mathcal{L}$. By induction, $\mathcal{L}^*$ admits an aspect ratio assignment α^* that is not weakly realizable. Augment α^* arbitrarily to an aspect ratio assignment α for $\mathcal{L}$. A weak realization of α on $\mathcal{L}$ would include a weak realization of α^* on $\mathcal{L}^*$, which is impossible, so $\mathcal{L}$ is not weakly ARU.

Case 2: all nontrivial sublayouts of $\mathcal{L}$ are sliceable. Then we can replace each maximal sublayout $\mathcal{L}^*$ of $\mathcal{L}$ with a rectangle r^* to obtain an irreducible layout $\mathcal{L}'$. By induction, there is an aspect ratio assignment α' for $\mathcal{L}'$ that is not realizable. By Lemma 6 there is a suitable

aspect ratio assignment α^* to each sliceable sublayout $\mathcal{L}^*$ such that the bounding box of its unique weak realization has the required aspect ratio $\alpha'(r^*)$. Thus the combination of α' and the assignment α^* yields an aspect ratio assignment for $\mathcal{L}$ that is not weakly realizable. ◄

We are now ready to prove Theorems 2 and 3. We restate both theorems for clarity.

▶ **Theorem 2.** *A generic layout is weakly aspect ratio universal if and only if it is sliceable.*

Proof. Let $\mathcal{L}$ be a generic layout. If $\mathcal{L}$ is sliceable, then it is weakly ARU by Corollary 7. Otherwise, it is not weakly ARU by Lemma 13. ◄

▶ **Theorem 3.** *For a generic layout $\mathcal{L}$, the following properties are equivalent:*
(i) $\mathcal{L}$ is strongly aspect ratio universal;
(ii) $\mathcal{L}$ is one-sided and sliceable;
(iii) the extended dual graph $G^*(\mathcal{L})$ of $\mathcal{L}$ admits a unique transversal structure.

Proof. Properties (ii) and (iii) are equivalent by Lemma 1. Property (ii) implies (i) by Corollary 8. For the converse, we prove the contrapositive. Let $\mathcal{L}$ be a layout that is not one-sided or not sliceable. If $\mathcal{L}$ is not sliceable, then it is not weakly ARU by Lemma 13, hence not strongly ARU, either. Otherwise $\mathcal{L}$ is sliceable but not one-sided, and then $\mathcal{L}$ is not strongly ARU by Lemma 10. ◄

2.2 Unique transversal structure

Subdividing a square into squares has fascinated humanity for ages [3, 14, 33]. For example, a *perfect square tiling* is a tiling with squares with distinct integer side lengths. Schramm [31] (see also [20, Chap. 6]) proved that every near-triangulation with an outer 4-cycle is the extended dual graph of a (possibly degenerate or nongeneric) subdivision of a rectangle into squares. The result generalizes to rectangular faces of arbitrary aspect ratios (rather than squares):

▶ **Theorem 14.** *(Schramm [31, Thm. 8.1]) Let $T = (V, E)$ be a near-triangulation with an outer 4-cycle, and $\alpha\colon V^* \to \mathbb{R}^+$ a function on the set V^* of the inner vertices of T. Then there exists a unique (but possibly degenerate or nongeneric) layout $\mathcal{L}$ such that $G^*(\mathcal{L}) = T$, and for every $v \in V^*$, the aspect ratio of the rectangle corresponding to v is $\alpha(v)$.*

The caveat in Schramm's result is that all rectangles in the interior of every separating 3-cycle must degenerate to a point, and rectangles in the interior of some of the separating 4-cycles may also degenerate to a point. We only use the *uniqueness* claim under the assumption that a nondegenerate and generic realization exists for a given aspect ratio assignment.

▶ **Lemma 15.** *If a layout $\mathcal{L}$ is strongly ARU, then its extended dual graph $G^*(\mathcal{L})$ admits a unique transversal structure.*

Proof. Consider the extended dual graph $T = G^*(\mathcal{L})$ of a strongly ARU layout $\mathcal{L}$. As noted above, T is a 4-connected inner triangulation of a 4-cycle. If T admits two different transversal structures, then there are two strongly nonequivalent layouts, $\mathcal{L}$ and $\mathcal{L}'$, such that $T = G^*(\mathcal{L}) = G^*(\mathcal{L}')$, which in turn yield two aspect ratio assignments, α and α', on the inner vertices of T. By Theorem 14, the (nondegenerate) layouts $\mathcal{L}$ and $\mathcal{L}'$ that realize α and α' are unique. Consequently, neither of them can be strongly aspect ratio universal. ◄

Lemma 15 readily shows that Theorem 3(i) implies Theorem 3(iii), and provides an alternative proof for the geometric arguments in Lemmata 10 and 13.

## 3   Recognizing dual graphs of aspect ratio universal layouts

In this section, we describe an algorithm that, for a given graph G, either finds a one-sided and sliceable (hence strongly ARU) layout $\mathcal{L}$ whose dual graph is G, or reports that no such layout exists (Theorem 4).

Assume that we are given a near-triangulation G (that is, a plane graph where every bounded face is a triangle). A slice of a layout corresponds to an edge cut in the dual graph that contains at most two edges of the outer face. If a slice is one-sided, the edges in the edge cut form a star; and the edge cut is determined by its edges on the boundary of the outer face. A brute force algorithm would guess a slice (i.e., edge cut), and recurse on the two subproblems; it would run in exponential time. We obtain a polynomial-time algorithm using a key insight: If we already know a slice, then in each subproblem, we know two corner rectangles. Our algorithm will utilize partial information about the corner rectangles.

Problem formulation.   The input of our recursive algorithm will be an *instance* $I = (G, C, P)$, where G is a near-triangulation, $C \colon V(G) \to \mathbb{N}_0$ is a *corner count*, and P is a set of ordered pairs (u, v) of vertices on the outer face of G. An instance $I = (G, C, P)$ is *realizable* if there exists a one-sided sliceable layout $\mathcal{L}$ such that G is the dual graph of $\mathcal{L}$, every vertex $v \in V(G)$ corresponds to a rectangle in $\mathcal{L}$ incident to *at least* $C(v)$ corners of the bounding box R of $\mathcal{L}$, and every pair $(a, b) \in P$ corresponds to a pair of rectangles in $\mathcal{L}$ incident to two counterclockwise *consecutive* corners of R. When we have no information about corners, then $C(v) = 0$ for every $v \in V$, and $P = \emptyset$. For convenience, we also maintain the total count $C(V) = \sum_{v \in V} C(v)$, and the set $K = \{v \in V(G) : C(v) > 0\}$ of vertices with positive corner count.

3.1   Structural properties of one-sided sliceable layouts

Throughout this section, we use a default notation: If an instance (G, C, P) is realizable by a one-sided sliceable layout $\mathcal{L}$, then R denotes the bounding box of $\mathcal{L}$, and for every $v \in V(G)$, r_v denotes the rectangular face of $\mathcal{L}$ corresponding to v. Note that any sublayout of a one-sided sliceable layout is also one-sided and sliceable (i.e., both properties are hereditary).

(a)   (b)   (c)   (d)
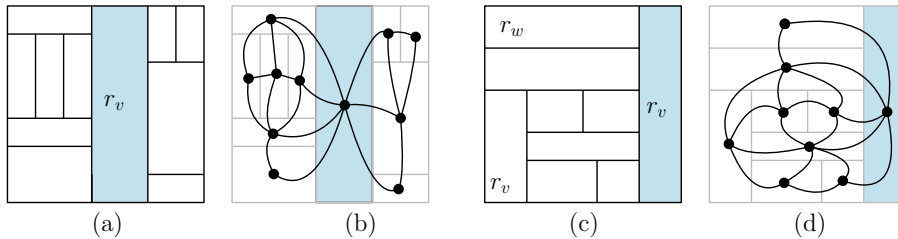
Figure 8 A one-sided sliceable layout and its dual graph G. (a–b) If v is a cut vertex of G, then r_v is bounded by two slices. (c–d) If there is no cut vertex in G, then some rectangular face r_v is incident to two corners.

▶ **Lemma 16.** *Assume that (G, C, P) admits a realization $\mathcal{L}$ and $|V(G)| \geq 2$. Then G contains a vertex v with one of the following (mutually exclusive) properties.*
(I) Vertex v is a cut vertex in G. Then r_v is bounded by two parallel sides of R and by two parallel slices; and $C(v) = 0$; see Fig. 8a–8b).
(II) Rectangle r_v is bounded by three sides of R and by a slice; and $0 \leq C(v) \leq 2$; see Fig. 8c–8d.

Proof. Let v be a cut vertex of G. Then r_v intersects the boundary of R in at least two disjoint arcs. Since both r_v and R are axis-parallel rectangles and $r_v \subset R$, their boundaries can intersect in at most two disjoint arcs, which are two parallel sides of r_v. The other two parallel sides of r_v form slices. In particular, r_v is bounded by two parallel sides of R and two slices, and so it is not incident to any corner of R. In this case, v has property (I).

Assume that G does not have cut vertices. Since $\mathcal{L}$ is sliceable, it is subdivided by a slice s which is a segment between two opposite sides of R. Since $\mathcal{L}$ is one-sided, s must be the side of a rectangle r_v for some $v \in V(G)$. If both sides of r_v parallel to s are in the interior of R, then r_v is bounded by two sides of R and by two slices. Since the sublayouts of $\mathcal{L}$ on the opposite sides of these slices are disjoint, then v is a cut vertex in G, contrarily to our assumption. Consequently, the other side of r_v parallel to s must be a side of R. Then r_v is bounded by three sides of R and by s. Clearly, r_v is incident to precisely two corners of R, and so v has property (II). ◀

Based on property (II), we say that a vertex v of G is a *pivot* if there exists a one-sided sliceable layout $\mathcal{L}$ with $G \simeq G(\mathcal{L})$ in which r_v is bounded by three sides of R and a slice. If we find a cut vertex or a pivot v in G, then at least one side of r_v is a slice, so we can remove v and recurse on the connected components of $G - v$.

Recursive calls. Given an instance $I = (G, C, P)$, we define the subproblems created by removing the vertex v in both cases (where v is either a cut vertex or a pivot):

(I) For a cut vertex v of G, we define the operation $\text{SPLIT}(G, C, P; v)$ as follows. The graph $G - v$ must have precisely two components, G_1 and G_2. Let $(u_1, \ldots, w_1)$ and $(u_2, \ldots, w_2)$ be the sequence of neighbors of v in G_1 and G_2, resp., in clockwise order. Initialize C_1 and C_2 as the restriction of C to $V(G_1)$ and $V(G_2)$, respectively. Set $C_i(u_i) \leftarrow C_i(u_i) + 1$ and $C_i(w_i) \leftarrow C_i(w_i) + 1$ for $i \in \{1, 2\}$ (if $u_i = w_i$, we increment $C_i(u_i)$ by 2). For each pair $(a, b) \in P$, if both a and b are in $V(G_i)$ for some $i \in \{1, 2\}$, then add (a, b) to P_i. Otherwise we may assume, without loss of generality, that $a \in V(G_1)$ and $b \in V(G_2)$, and the counterclockwise path (a, b) contains either u_1, v, w_2 or w_1, v, u_2. The removal of v splits the path into two subpaths, that we add into P_1 and P_2, accordingly. Finally we add (u_1, w_1) to P_1 and (u_2, w_2) to P_2. Return the instances (G_1, C_1, P_1) and (G_2, C_2, P_2).

▶ **Lemma 17.** *Let v be a cut vertex of G. An instance (G, C, P) is realizable if and only if both instances returned by $\text{SPLIT}(G, C, P; v)$ are realizable.*

Proof. First assume that $\mathcal{L}$ is a realization of instance (G, C, P). The removal of rectangle r_v splits $\mathcal{L}$ into two one-sided and sliceable sublayouts, $\mathcal{L}_1$ and $\mathcal{L}_2$. It is easily checked that they realize (G_1, C_1, P_1) and (G_2, C_2, P_2), respectively.

Conversely, if both (G_1, C_1, P_1) and (G_2, C_2, P_2) are realizable, then they are realized by some one-sided sliceable layouts $\mathcal{L}_1$ and $\mathcal{L}_2$, respectively. The union of a square r_v and scaled copies of $\mathcal{L}_1$ and $\mathcal{L}_2$ attached to two opposite sides r_v yields a one-sided sliceable layout $\mathcal{L}$ that realizes (G, C, P). ◀

(II) Let v be a pivot of G. We define the operation $\text{REMOVE}(G, C, P; v)$ as follows. Since v is not a cut vertex, then $G - v$ has precisely one component, which we denote by G'. Let $(u, \ldots, w)$ be the sequence of neighbors of v in G' in clockwise order. Initialize C' as the restriction of C to $V(G')$, and then set $C'(u) \leftarrow C'(u) + 1$ and $C'(w) \leftarrow C'(w) + 1$. If for any pair $(a, b) \in P$, the counterclockwise path from a to b in G contains v, then return FALSE. Otherwise, set $P' = P$, and add (u, w) to P'. Return the instance (G', C', P').

▶ **Lemma 18.** *Let v be a vertex of the outer face of G, but not a cut vertex. An instance (G, C, P) is realizable with pivot v if and only if the instance returned by $\textsc{Remove}(G, C, P; v)$ is realizable.*

Proof. Assume that (G, C, P) is realized by a one-sided scliceable layout $\mathcal{L}$, and v has property (II). The removal of rectangle r_v from $\mathcal{L}$ creates a one-sided sliceable sublayout $\mathcal{L}'$. It is easily checked that $\mathcal{L}'$ realizes the instance (G', C', P') returned by $\textsc{Remove}(G, C, P; v)$.

Conversely, assume that the instance (G', C', P') returned by $\textsc{Remove}(G, C, P; v)$ is realized by a layout $\mathcal{L}'$. Then let s_{uw} be the maximum segment that connects the two corners of the bounding box of $\mathcal{L}'$ incident to the r_u and r_w, where $(u, w) \in P' \setminus P$. We can attach a single rectangular face r_v to the bounding box of $\mathcal{L}'$ along segment s_{uw} and obtain a layout $\mathcal{L}$ that realizes (G, C, P). ◀

Finding a pivot. In the absence of a cut vertex, any vertex incident to the outer face of G might be a pivot. We use partial information on the corners to narrow down the search for a pivot.

▶ **Lemma 19.** *Assume that an instance (G, C, P) admits a realization $\mathcal{L}$. If $|V(G)| \geq 2$ and $C(v) \geq 2$ for some vertex $v \in V(G)$, then v is a pivot.*

Proof. The rectangle r_v is incident to at least two corners of R. If r_v is incident to two opposite corners of R, then $r_v = R$, contradicting the assumption that G has two or more vertices. Hence r_v is incident to two consecutive corners of R, and so it contains some side s of R. The other side of r_v parallel to s is a maximal segment between two opposite sides of R, so it must be a slice. ◀

In the absence of a cut vertex, we are looking for a pivot in a 2-vertex-cut (i.e., 2-cut). Recall that a sliceable layout $\mathcal{L}$ corresponds to a BSP-tree in which every nonleaf node corresponds to a sublayout $\mathcal{L}'$ of $\mathcal{L}$ together with a slice of $\mathcal{L}'$.

▶ **Lemma 20.** *Let $\mathcal{L}$ be a one-sided sliceable layout whose dual graph $G(\mathcal{L})$ is 2-connected and has a 2-cut $\{u, v\}$. Then there exists a one-sided sliceable layout $\mathcal{L}'$ such that $G(\mathcal{L}') \simeq G(\mathcal{L})$ and the root slice separates the rectangles corresponding to u and v. In particular, u or v is a pivot.*

Furthermore, if there are two rectangular faces in $\mathcal{L}$ that are each incident to exactly one corner of $\mathcal{L}$, then the corresponding rectangular faces in $\mathcal{L}'$ are also incident to some corners in $\mathcal{L}'$, or there exists a one-sided sliceable layout $\mathcal{L}''$ with $G(\mathcal{L}'') \simeq G(\mathcal{L})$ in which one of these rectangular faces is a pivot.

Proof. Let R be the bounding box of $\mathcal{L}$; let r_u and r_v denote the rectangular faces corresponding to u and v, respectively. Since G is a near-triangulation, then u and v are adjacent in $V(G)$. Let s_0 be the maximal segment that separates r_u and r_v. If s_0 connects two opposite sides of R, the proof is complete with $\mathcal{L}' = \mathcal{L}$, so we may assume otherwise. Because $\mathcal{L}$ is one-sided, whenever two rectangular faces are in contact, a side of one rectangle fully contains a side of the other. We distinguish between two cases:

Case 1: r_u and r_v are in contact with opposite sides of R. We may assume, without loss of generality, that r_u and r_v are in contact with the bottom and top side of R, resp., and the bottom side of r_v contains the top side of r_u; see Fig. 9a. Since $\mathcal{L}$ is one-sided, s_0 is a side of some rectangular face in $\mathcal{L}$, and we may assume that s_0 is the bottom side of r_v. The left (resp., right) side of r_v lies either along ∂R or in a vertical segments s_1 (resp., s_2). Since s_0 does not reach both left and right sides of R by assumption, at least one of

s_1 and s_2 exists. Assume, without loss of generality, that s_1 exists. Since the bottom-left corner of r_v is the endpoint of s_0, it lies in the interior of segment s_1.

We claim that the bottom endpoint of s_1 is on the bottom side of R. Suppose otherwise. Then we can incrementally construct a *counterclockwise winding path* W: it starts with a downward edge along s_1. Append an edge to W by making a left turn and following a maximal segment to its endpoint, until the endpoint of W is on the boundary of R or in the interior of a previous edge of W. As W can cross neither r_u nor r_v, and it cannot make a right turn if it reaches s_0, then it cannot reach the boundary of R. Consequently, W ends in the interior of one of its previous edges, and the maximal segments generated by its last four edges form a counterclockwise windmill, contradicting the assumption that $\mathcal{L}$ is sliceable. This completes the proof of the claim.
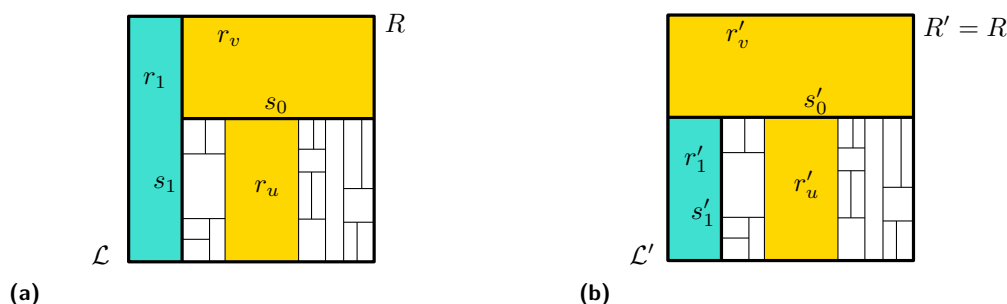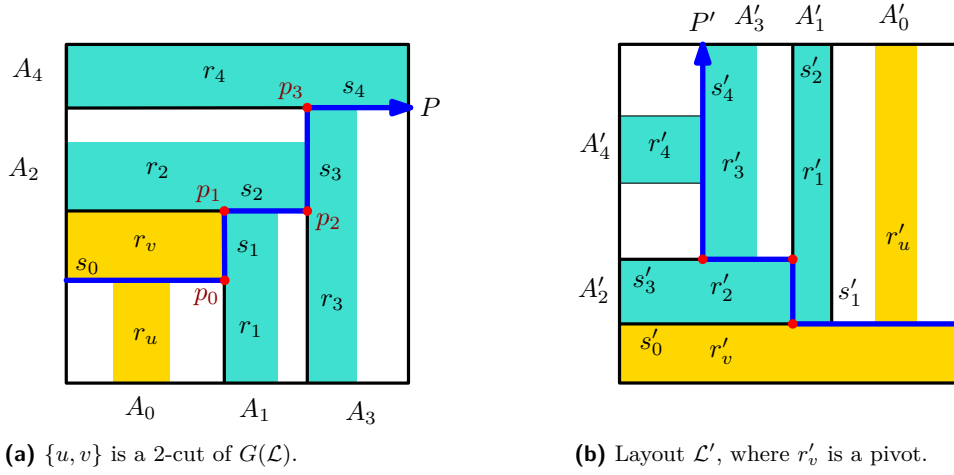

Figure 9 (a) A one-sided sliceable layout $\mathcal{L}$ where r_u and r_v touch two opposite sides of the bounding box R. (b) The modified layout $\mathcal{L}'$ within the same bounding box $R' = R$.

We can now modify $\mathcal{L}$ by extending s_0 and r_v horizontally to the right side of R, and clip both s_1 and r_1 to s_0, as in Fig. 9b. This modification changes the contacts between r_v and r_1 from vertical to horizontal, but does not change any other contacts in the layout, so it does not change the dual graph $G(\mathcal{L})$. If segment s_2 exists, we can similarly extend s_0 to the right side of R and clip s_2. We obtain a sliceable layout $\mathcal{L}'$ with $G(\mathcal{L}') \simeq G(\mathcal{L})$ in which the root slice s_0' separates r_u' and r_v', and rectangle r_v' is a pivot. Furthermore, every rectangle incident to a corner in $\mathcal{L}$ remains incident to some corner in $\mathcal{L}'$.

Case 2: r_u and r_v are not in contact with opposite sides of R. Then r_u and r_v are each in contact with a single side of R, and these sides are adjacent. We may assume, without loss of generality, that r_u is in contact with the bottom side of R, r_v is in contact with the left side of R, and the bottom side of r_u contains the top side of r_v; refer to Fig. 10a. Because $\mathcal{L}$ is one-sided and s_0 is not the root slice, then s_0 equals the bottom side of r_v.

We incrementally construct an x- and y-monotone increasing directed path P (i.e., a staircase) starting with edge s_0, directed to its right endpoint p_0. Initially, let $P = \{s_0\}$ and $i := 0$. While p_i is not in the top or right side of R, let p_{i+1} be the top or right endpoint of segment s_i, append the edge $p_i p_{i+1}$ to P, and let s_{i+1} be the segment orthogonal to s_i that contains p_{i+1}. Since the path P is x- and y-monotonically increasing, it does not revisit any segment. Thus the recursion terminates, and P reaches the top or right side of R.

Let s_{-1} denote the right side of r_u. We claim that for $i = 1, \ldots, k$, if s_i is vertical, its bottom endpoint is on the bottom side of R, and if s_i is horizontal, its left endpoint is on the left side of R. The claim clearly holds for $i = 0$, since s_0 is the bottom side of r_v, which is in contact with the left side of R. Suppose for contradiction that the claim holds for s_{i-1} but not for s_i. Then the clockwise or counterclockwise winding path starting with s_i would create a windmill (as it can cross neither s_{i-1} nor s_{i-2}, contradicting the assumption that $\mathcal{L}$ is sliceable.

(a) $\{u, v\}$ is a 2-cut of $G(\mathcal{L})$.

(b) Layout $\mathcal{L}'$, where r'_v is a pivot.

■ **Figure 10** The construction of $\mathcal{L}'$ from $\mathcal{L}$.

The segments $s_0, s_1, \ldots, s_k$ jointly form a one-sided sliceable layout, that is, they subdivide R into $k + 2$ rectangular regions, each of which contains a sublayout of $\mathcal{L}$. One of these regions is r_v. Label the remaining k regions by $A_0, A_1, \ldots, A_k$ in the order in which they occur along P; see Fig. 10a. In particular, we have $r_u \subset A_0$. For $i = 1, \ldots, k$, region A_i is bounded by ∂R and segments s_i, s_{i+1}, and s_{i+2} (if they exist); and A_k is adjacent to the top-right corner of R. Because $\mathcal{L}$ is one-sided, segment s_i is a side of a rectanglular face that we denote by r_i, for $i = 1, \ldots, k$; and $r_i \subseteq A_i$ as the opposite side of s_i is subdivided by segment s_{i-1}.

Furthermore, we claim that $A_k = r_k$. Indeed, A_k is bounded by segment s_k and three sides of B. If $A_k \neq r_k$, then r_k separates the subarrangement $A \setminus r_k$ from r. This means that v_{r_k} would be a cut vertex in $G(\mathcal{L})$, contradicting the assumption that $G(\mathcal{L})$ is 2-connected.

We recursively construct a one-sided sliceable $\mathcal{L}'$ by placing rectangles and subarrangement corresponding to those in $\mathcal{L}$ such that $G(\mathcal{L}) \simeq G(\mathcal{L}')$; refer to Fig. 10b. Let R' be the bounding box of $\mathcal{L}'$. First subdivide R' by a horizontal segment s'_0; and let r'_v be the rectangle below s'_0. This ensures that s'_0 is the root slice and r'_v is a pivot. Subdivide the region above s'_0 by a vertical segment s'_1 into two rectangular regions. Denote the right region by A'_0, and subdivide the left region as follows: For $i = 2, \ldots, k$, recursively subdivide the rectangle incident to the top-left corner of R' by a segment s'_i orthogonal to s_i.

Segments $s'_0, \ldots, s'_k$ jointly subdivide R' into $k+2$ rectangular regions: r'_v and $A'_0, A'_1, \ldots, A'_k$ in the order in which they are created, where A'_k is incident to the top-left corner of R'; and all other regions are in contact with either the left or the top side of R'. We insert a sublayout in each region A'_i. First insert a 180°-rotated affine copy of A_0 into A'_0. For $i = 1, \ldots, k - 1$, insert r'_i into A'_i such that its top or left side is s'_{i+1}; and if $A_{i-2} \setminus r_{i-2}$ is nonempty, insert an affine copy of the sublayout $A_{i-1} \setminus r_{i-1}$ into A'_k. Finally, for $i = k$, subdivide A'_k into three rectangles by slices orthogonal to s'_k: If $A_{k-2} \setminus r_{k-2}$ or $A_{k-1} \setminus r_{k-1}$ is nonempty, insert an affine copy in the first and third rectangle in A'_k; and fill all remaining space by r'_k. This completes the construction of layout $\mathcal{L}'$ (see Fig. 10b). By construction, we have $G(\mathcal{L}') \simeq G(\mathcal{L})$.

It remains to track the rectangles incident to the corners of $\mathcal{L}$ and $\mathcal{L}'$. In the original layout $\mathcal{L}$, the rectangular face r_k is incident to two corners of R. Assume, without loss of generality, that r_k is incident to the two top corners of R (as in Fig. 10a), and two

distinct rectangular faces $r_{\text{left}} \subset A_0$ and $r_{\text{right}} \subset A_{k-1}$ are incident to the bottom-left and bottom-right corners of R, respectively. The sublayout A_0 was inserted into A_0' after a $180°$ rotation, and so $r_{\text{left}}' \subset A_0'$ is incident to the top-right corner in $\mathcal{L}'$. If $r_{\text{right}} \subset A_{k-1} \setminus r_{k-1}$, then $A_{k-1} \setminus r_{k-1}$ is nonempty and it was inserted into the top third of A_k' after a $180°$ rotation, and so r_{right}' is incident to the top-left corner in $\mathcal{L}'$. Otherwise $A_{k-1} = r_{k-1}$, and then $r_{\text{right}} = r_{k-1}$. In this case r_k' is incident to the top-left corner in $\mathcal{L}'$. However, we can modify $\mathcal{L}$ by extending r_{k-1} and s_{k-1} to the top side of R, and obtain a one-sided sliceable layout $\mathcal{L}''$ with $G(\mathcal{L}') \simeq G(\mathcal{L})$ in which $r_{\text{right}}'' = r_{k-1}''$ is a pivot. This completes the proof in Case 2. ◀

▶ **Lemma 21.** *Assume that an instance (G, C, P) is realizable; G is 2-connected; $|V(G)| \geq 4$; there exist two adjacent vertices, u and v, such that $C(u) = C(v) = 1$, and $C(w) = 0$ for all other vertices. Then u or v is a pivot; or else G has a 2-cut and a vertex of an arbitrary 2-cut is a pivot.*

Proof. Let $\mathcal{L}$ be a one-sided sliceable layout that realizes (G, C, P). If r_u and r_v contain two opposite corners of $\mathcal{L}$, then the maximal line segment that separates them is a slice of $\mathcal{L}$, and so r_u or r_v must also contain another corner, and thus be a pivot.

We may assume, then, that r_u and r_v contain adjacent corners of $\mathcal{L}$, which we may assume, without loss of generality, to be the top-left and bottom-left corners, respectively. If either spans the width of $\mathcal{L}$ and contains another corner, then it corresponds to a pivot and the proof is complete.
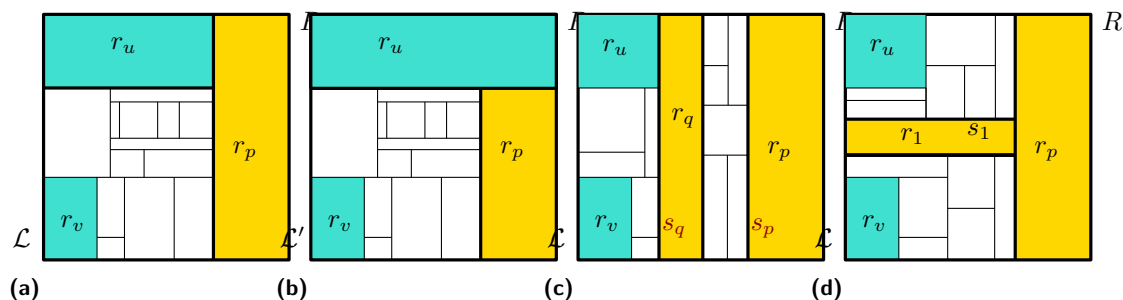
■ **Figure 11** (a) Rectangles r_u, r_u, and r_p jointly contain all four corners of R. (b) Modifying the boundary between r_u and r_p. (c) If both s_p and s_q are slices, then q would be a cut vertex. (d) The topmost segment s_1 between the left edge of R and the left edge of r_p is a side of r_1.

Assume that r_u and r_v each contain only one corner of $\mathcal{L}$; see Fig. 11. There is some rectangular face r_p that contains the top-right and bottom-right corners of $\mathcal{L}$, or else there would be no pivot, contradicting Lemma 16. If r_u and r_p are adjacent, then we can modify $\mathcal{L}$ by removing the right side of r_u and extending the bottom side of r_u to the right side of R; refer to Figs. 11a–11b. This yields a layout $\mathcal{L}'$ realizing (G, C, P) in which r_u contains two corners, and thus u is a pivot. The same argument can be made for v being a pivot if r_v and r_p are adjacent.

In the remainder of the proof, we assume that neither r_u nor r_v is in contact with r_p. The left side of r_p is a slice as it connects the top and bottom sides of R. Note that R cannot have any other slice, as it would be a side of some rectangle r_q, which does not contain any corner (cf. Fig. 11c); and so q would be a cut vertex, contrary to out assumption that G is 2-connected. This implies, using the fact that $\mathcal{L}$ is sliceable, that $\mathcal{L}$ has at least one horizontal segment from the left side of R to the left side of r_p. Let s_1 be the topmost such segment; see Fig. 11d. As $\mathcal{L}$ is one-sided, then s_1 must be the side of some rectangular face

r_1. The rectangular face r_1 can be neither r_u nor r_v, since they are not in contact with r_p. The vertices in G corresponding to r_1 and r_p form a 2-cut, so G has a 2-cut. Lemma 20 guarantees that for any 2-cut, there is a one-sided sliceable layout $\mathcal{L}'$ with $G(\mathcal{L}') \simeq G(\mathcal{L})$ in which one of the vertices of the 2-cut is a pivot. Such a layout $\mathcal{L}'$ realizes (G, C, P) if the rectangular faces corresponding to u and v are each incident to some corners of $\mathcal{L}'$. Since r_u and r_v are each incident to a single corner of $\mathcal{L}$, then Lemma 20 further guarantees that $\mathcal{L}'$ is a one-sided sliceable layout that realizes (G, C, P) and one of the vertices in the 2-cut is a pivot; or else u or v is a pivot. ◀

Recall that for an instance $I = (G, C, P)$, we defined the set $K = \{v \in V(G) : C(v) > 0\}$ of vertices that must contain corners in the realization. We show that if we already have 3 or 4 vertices in K, then G has a cut vertex or a vertex in K is a pivot.

▶ **Lemma 22.** *Assume that (G, C, P) is realizable and $|V(G)| \geq 2$.*
1. *If $|K| = 4$, then G has a cut vertex.*
2. *If $|K| = 3$, then G has a cut vertex or some vertex $v \in K$ is a pivot.*

Proof. Let $\mathcal{L}$ be a realization of (G, C, P). If G has a cut vertex, the proof is complete. Assume otherwise. Then, by Lemma 16, there exists a rectangular face r_u is incident to two corners of $\mathcal{L}$, and in particular $u \in V(G)$ is a pivot. As R has only four corners, each of which is incident to a unique rectangular face in $\mathcal{L}$, then at most two additional rectangular faces in $\mathcal{L}$ are incident to corners, hence $|K| \leq 3$.

Assume that $|K| = 3$. Since R has only four corners, each of which is incident to a rectangular face in $\mathcal{L}$, the pivot u is one of the three vertices in K. ◀

3.2   Recognition algorithm

We are now ready to prove the main result of this section.

▶ **Theorem 4.** *We can decide in $O(n^2)$ time whether a given graph G with n vertices is the dual of a one-sided sliceable layout.*

Proof. Given a graph G, we can decide in $O(n)$ time whether G is a proper graph [12, 23, 25, 26]. If G is proper, then it is a connected plane graph in which all bounded faces are triangles. Let an initial instance be $I = (G, C, P)$, where $C(v) = 0$ for all vertices v, and $P = \emptyset$. We run the recursive algorithm RECOGNIZE(G, C, P) below.

Correctness. We argue that algorithm RECOGNIZE(G, C, P) correctly reports whether an instance (G, C, P) is realizable.

Lines 3–4. A graph with only one vertex corresponds to a layout containing a single rectangle, which is clearly aspect ratio universal.

Lines 5–6. A rectangle that contains three or more corners of a layout must be the only rectangle in the layout. However, the algorithm reaches this step only if there are multiple vertices in the graph (Lines 3–4), so a vertex with a corner count of 3 or more is a contradiction.

Lines 7–9. Lemma 17 established that an instance is realizable if and only if both instances produced by a SPLIT operation are realizable.

Lines 10–26. In the absence of a cut vertex, we try to find a pivot. Lemma 18 established that an instance is realizable if and only if it remains realizable after removing a pivot.

(A) **Lines 10–11.** Lemma 19 shows that a vertex v with $C(v) = 2$ is a pivot.

1 RECOGNIZE(G, C, P)

> **input :** a near triangulation G, a corner count $C\colon V(G) \to \mathbb{N}_0$, and a set P of ordered pairs of vertices on the outer face of G
>
> **output :** whether G has an aspect ratio universal dual

2 begin

3   **if** $|V(G)| = 1$ **then**

4   **return** *True*

5   **else if** *G has a vertex v with* $C(v) > 2$ **then**

6   **return** *False*

7   **else if** *G has a cut vertex v* **then**

8   SPLIT$(G, C, P; v)$ yields (G_1, C_1, P_1) and (G_2, C_2, P_2)

9   **return** RECOGNIZE$(G_1, C_1, P_1) \wedge$ RECOGNIZE(G_2, C_2, P_2)

10   **else if** *G has a vertex v with* $C(v) = 2$ **then**

11   **return** RECOGNIZE$(\text{REMOVE}(G, C, P; v))$

12   **else if** $P = \{(u, v)\}$ *with* $C(u) = C(v) = 1$ *and* $|K| = 2$ **then**

13   **foreach** $w \in \{u, v\}$ **do**

14   **if** RECOGNIZE$(\text{REMOVE}(G, C, P; w))$ **then**

15   **return** *True*

16   **if** *G has a 2-cut* **then**

17   Let $\{w_1, w_2\}$ be an arbitrary 2-cut of G

18   **foreach** $w \in \{w_1, w_2\}$ **do**

19   **if** RECOGNIZE$(\text{REMOVE}(G, C, P; w))$ **then**

20   **return** *True*

21   **else if** $|K| = 3$ **then**

22   **foreach** $v \in K$ **do**

23   **if** RECOGNIZE$(\text{REMOVE}(G, C, P; v))$ **then**

24   **return** *True*

25   **else if** $|K| = 0$ **then**

26   **foreach** *vertex v of the outer face of G* **do**

27   **if** RECOGNIZE$(\text{REMOVE}(G, C, P; v))$ **then**

28   **return** *True*

29   **return** *False*

(B) **Lines 12–20.** Lemma 21 shows that if $C(u) = C(v) = 1$ and the corner count of all other vertices is zero (hence $K = \{u, v\}$ and $|K| = 2$), then u or v is a pivot, or at least one vertex in every 2-cut is a pivot. The algorithm tests whether u or v is a pivot, or any vertex of an arbitrary 2-cut is a pivot.

(D) **Lines 21–24.** By Lemma 22, when $|K| = 3$, a vertex in K must be a cut vertex or a pivot, or else the instance is not realizable.

(E) **Lines 25–28.** If we have no information about the corners and there is no cut vertex (that is, $C(v) = 0$ for all $v \in V(G)$, hence $K = \emptyset$), then one of the vertices in the outer face must correspond to a pivot by Lemma 16, or else (G, C, P) is not realizable.

Line 29. If we find neither a cut vertex nor a pivot, then the instance is not realizable by Lemma 18.

Data structures. Algorithm $\text{RECOGNIZE}(G, C, P)$ recursively removes a cut vertex or a pivot v of G, and recurses on connected components of $G - v$. The recursive calls are supported by an in-place data structure that dynamically maintains the instance (G, C, P). Specifically, we maintain G in the classical *doubly-link edge list* (for short *DCEL*) data structure [11, 35]; see also [16]. It maintains the incidences between vertices, edges, and faces; and supports edge deletion in $O(1)$ time, hence the deletion of a vertex v in $O(\deg(v))$ time.

We augment the classical DCEL data structure with three binary indicator variables for (1) vertices of the outer face, (2) cut vertices, and (3) edges corresponding to 2-cuts. The DCEL data structure already maintains the cyclic list of vertices incident to the outer face. The algorithm maintains the property that G is an internally triangulated plane graph. This in turn implies that a vertex v is a cut vertex if and only if the outer face appears at least twice in the cyclic order of all faces incident to v. Similarly, $\{u, v\}$ is a 2-cut if and only if uv is an edge and the outer face appears at least twice in the cyclic order of all faces incident to uv. Consequently, we can identify new cut vertices (resp., edges corresponding to 2-cuts) in $O(1)$ time whenever a new vertex or edge becomes incident to the outer face. The SPLIT and REMOVE operations each remove a cut vertex or a pivot, which is incident to the outer face, and so the distance of any vertex to the outer face monotonically decreases. This implies that each indicator variable changes at most once, hence all three variables can be maintained in $O(n)$ total time in any descending path of the recursion tree.

Runtime analysis. Let T be the recursion tree of the algorithm $\text{RECOGNIZE}(G, C, P)$ for some initial instance (G, C, P). The number of vertices in G strictly decreases along each descending path of T, and so the depth of the tree is $O(n)$.

We distinguish between two types of nodes in T: If a step in lines 8–9 is executed, then the vertex set $V(G)$ is partitioned among the recursive subproblems; we call these *partition nodes* of T. In the steps in (B) lines 12–20, (D) lines 21–24, and (E) lines 25–28, however, $|V(G)| - 1$ vertices appear in all four, three, or $O(|V(G)|)$ recursive subproblems; we call these *duplication nodes* of T. The algorithm performs a DFS traversal of T sequentially, and so the same in-place DCEL data structure of size $O(n)$ can support the entire course of the algorithm.

We first analyze the special case that T does not have duplication nodes. Then T is a binary tree with $O(n)$ nodes. Overall, the total time taken by maintaining G (in the DCEL data structure) and the annotation C and P is $O(n)$ over the course of the algorithm.

Next, we analyze the impact of duplication nodes. We claim that steps (B), (D) and (E) are reached at most once. Note first that the total corner count $C(V) = \sum_{v \in V} C(v)$ monotonically increases along any descending path of T. In the initial instance, we have

$C(V) = 0$. All REMOVE and SPLIT operations produce subproblems with $C(V) \geq 2$. Every REMOVE operation removes one vertex v with $C(v) \leq 2$ and increments the total corner count by two.

Due to the steps in (A) lines 10-11, we may assume that $C(v) \in \{0, 1\}$ for all $v \in V(G)$, hence $C(V) = |K|$ in steps (B), (D), and (E). Step (B) can be reached only when $C(V) = 2$ and two distinct vertices have positive corner counts. In this case, operation REMOVE($G, C, P; v$) in line 14 or 19 removes at most one of these vertices, and increments the corner counts by two, yielding a subproblem with $C(V) = 3$. Similarly, step (D) can be reached only when $C(V) = 3$, and produces a subproblem with $C(V) = 4$. Finally, step (E) can be reached when $C(V) = 0$, and REMOVE($G, C, P; v$) in line 27 produces a yields a subproblem with $C(V) = 2$. As steps (B), (D), and (E) produce at most four, three, and $O(|V(G)|)$ recursive subproblems, resp., the duplication nodes increase the upper bound on the runtime by a factor of $4 \cdot 3 \cdot n = 12n$, hence it is $O(n^2)$. ◀

4 Conclusions

We have shown that a layout $\mathcal{L}$ is weakly (resp., strongly) ARU if and only if $\mathcal{L}$ is sliceable (resp., one-sided and sliceable); and one can decide in $O(n^2)$ time whether a given graph G on n vertices is the dual graph of a one-sided sliceable layout. An immediate open problem is whether the runtime can be improved. Recall that no polynomial-time algorithm is currently known for recognizing the dual graphs of sliceable layouts [4, 19, 38] and one-sided layouts [6]. It remains open to settle the computational complexity of these problems.

Cut vertices and 2-cuts play a crucial role in our algorithm. We can show (Proposition 23 below) that the dual graphs of one-sided sliceable layouts have vertex cuts of size at most three. In contrast, the minimum vertex cut in the dual graphs of one-sided layouts (resp., sliceable layouts) is unbounded. Perhaps 3-cuts can be utilized to speed up our algorithm.

▶ **Proposition 23.** *Let G be the dual graph of a one-sided sliceable layout. If $|V(G)| \geq 4$, then G contains a vertex cut of size at most 3.*

Proof. Let $\mathcal{L}$ be a one-sided sliceable layout with $n \geq 4$ rectangular faces in a bounding box R, and with dual graph $G = G(\mathcal{L})$. For a rectangular face r in $\mathcal{L}$, let $v(r)$ denote the corresponding vertex in G. If G is outerplanar, then either G has a cut vertex, or G is a triangulated n-cycle, hence any diagonal forms a 2-cut. We may assume that G has an interior vertex.

Consider a sequence of segments that recursively subdivide R into the layout $\mathcal{L}$; and let us focus on the first subdivision step that creates a rectangle R_0 in the interior of R. We may assume, without loss of generality, that R_0 is bounded by the segments $s_1, \ldots, s_4$ in counterclockwise order, the subdivision along s_4 splits some rectangle into R_0 and R_1, and s_4 is the bottom side of R_0. Then the two endpoints of s_4 lie in the relative interiors of s_1 and s_3. Since $\mathcal{L}$ is one-sided, s_1 is the right side of some rectangular face r_1, and s_3 is the left side of some rectangular face r_3 of $\mathcal{L}$. Since R_0 is the first rectangle in the interior of R, both s_1 and s_2 has an endpoint on the boundary of R; see Fig. 12a.

We claim that at most two maximal horizontal segments in $\mathcal{L}$ intersect both s_1 and s_3 in the interior of R. Suppose, to the contrary, that three or more such segments (Fig. 12b). Let s' be one of them other than the lowest or highest. Since $\mathcal{L}$ is one-sided, s' is a side of some rectangular face r' in $\mathcal{L}$, which is adjacent to both r_1 and r_3, but not adjacent to the boundary of R, hence $\{v(r_1), v(r'), v(r_3)\}$ is a 3-cut in G.

In the remainder of the proof, we may assume that at most two maximal horizontal segments intersect both s_1 and s_3 in the interior of R. Consequently s_2 (s_4) is the highest
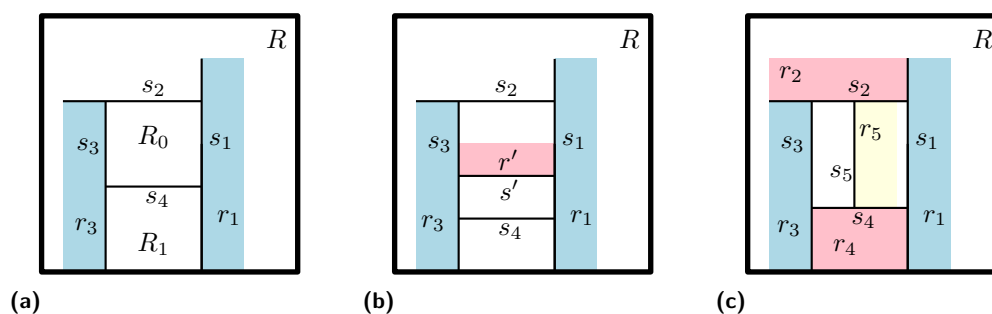
Figure 12 Schematic views of the arrangements in the proof of Proposition 23.

(lowest) such segment. Since $\mathcal{L}$ is one-sided, s_4 is the side of some rectangular face r_4 of $\mathcal{L}$. If s_4 is the bottom side of r_4, then $\{v(r_1), v(r_2), v(r_3)\}$ is a 3-cut in G. We may assume that s_4 is the top side of a rectangle r_4 in $\mathcal{L}$. Since no segment intersects both s_1 and s_3 below s_4, then $r_4 = R_1$ (as in Fig. 12c).

If R_0 is not sliced further recursively, then $\{v(r_1), v(R_0), v(r_3)\}$ is a 3-cut in G. Otherwise, let s_5 be the first segment that slices R_0. Segment s_5 cannot be horizontal, as it would intersect both s_1 and s_3, contrary to the assumption above. So s_5 is vertical (Fig. 12c), and it is a side of some rectangular face r_5 of $\mathcal{L}$, which is adjacent to s_2 and s_4. This further implies that s_2 is the bottom side of a rectanglular face r_2 of $\mathcal{L}$. If r_2 is adjacent to the boundary of R, then $\{v(r_4), v(r_5), v(r_2)\}$ is a 3-cut; else r_2 lies in the interior of R and $\{v(r_1), v(r_2), v(r_3)\}$ is a 3-cut in G. ◀

───── **References** ─────

1 Eyal Ackerman, Gill Barequet, and Ron Y. Pinter. A bijection between permutations and floorplans, and its applications. *Discret. Appl. Math.*, 154(12):1674–1684, 2006. `doi:10.1016/j.dam.2006.03.018`.

2 Andrei Asinowski and Toufik Mansour. Separable d-permutations and guillotine partitions. *Annals of Combinatorics*, 14:17–43, 2010. `doi:10.1007/s00026-010-0043-8`.

3 Rowland Leonard Brooks, Cedric A. B. Smith, Arthur H. Stone, and William T. Tutte. The dissection of rectangles into squares. *Duke Math. J.*, 7(1):312–340, 1940. `doi:10.1215/S0012-7094-40-00718-9`.

4 Parthasarathi Dasgupta and Susmita Sur-Kolay. Slicible rectangular graphs and their optimal floorplans. *ACM Trans. Design Autom. Electr. Syst.*, 6(4):447–470, 2001. `doi:10.1145/502175.502176`.

5 Mark de Berg, Otfried Cheong, Marc van Kreveld, and Mark Overmars. Binary space partitions. In *Computational Geometry: Algorithms and Applications*, chapter 12, pages 259–281. Springer, 2008. `doi:10.1007/978-3-540-77974-2\_12`.

6 David Eppstein, Elena Mumford, Bettina Speckmann, and Kevin Verbeek. Area-universal and constrained rectangular layouts. *SIAM J. Comput.*, 41(3):537–564, 2012. `doi:10.1137/110834032`.

7 Stefan Felsner. Lattice structures from planar graphs. *Electron. J. Comb.*, 11(1), 2004. URL: `http://www.combinatorics.org/Volume_11/Abstracts/v11i1r15.html`.

8 Stefan Felsner. Rectangle and square representations of planar graphs. In János Pach, editor, *Thirty Essays on Geometric Graph Theory*, pages 213–248. Springer, 2013. `doi:10.1007/978-1-4614-0110-0_12`.

9 Stefan Felsner. Exploiting air-pressure to map floorplans on point sets. *J. Graph Algorithms Appl.*, 18(2):233–252, 2014. `doi:10.7155/jgaa.00320`.

10 Éric Fusy. Transversal structures on triangulations: A combinatorial study and straight-line drawings. *Discret. Math.*, 309(7):1870–1894, 2009. `doi:10.1016/j.disc.2007.12.093`.

11 Leonidas J. Guibas and Jorge Stolfi. Primitives for the manipulation of general subdivisions and the computation of Voronoi diagrams. *ACM Trans. Graph.*, 4(2):74–123, 1985. `doi:10.1145/282918.282923`.

12 Md. Manzurul Hasan, Md. Saidur Rahman, and Muhammad Rezaul Karim. Box-rectangular drawings of planar graphs. *J. Graph Algorithms Appl.*, 17(6):629–646, 2013. `doi:10.7155/jgaa.00309`.

13 Xin He. On finding the rectangular duals of planar triangular graphs. *SIAM J. Comput.*, 22(6):1218–1226, 1993. `doi:10.1137/0222072`.

14 Frederick V. Henle and James M. Henle. Squaring the plane. *Am. Math. Mon.*, 115(1):3–12, 2008. URL: `http://www.jstor.org/stable/27642387`.

15 Goos Kant and Xin He. Regular edge labeling of 4-connected plane graphs and its applications in graph drawing problems. *Theor. Comput. Sci.*, 172(1-2):175–193, 1997. `doi:10.1016/S0304-3975(95)00257-X`.

16 Lutz Kettner. *Software design in computational geometry and contour-edge based polyhedron visualization.* PhD thesis, ETH Zürich, 1999. `doi:10.3929/ethz-a-003861002`.

17 Krzysztof Koźmiński and Edwin Kinnen. Rectangular duals of planar graphs. *Networks*, 15(2):145–157, 1985. `doi:10.1002/net.3230150202`.

18 Marc J. van Kreveld and Bettina Speckmann. On rectangular cartograms. *Comput. Geom.*, 37(3):175–187, 2007. `doi:10.1016/j.comgeo.2006.06.002`.

19 Vincent Kusters and Bettina Speckmann. Towards characterizing graphs with a sliceable rectangular dual. In *Proc. 23rd Sympos. Graph Drawing and Network Visualization (GD)*, volume 9411 of *LNCS*, pages 460–471. Springer, 2015. `doi:10.1007/978-3-319-27261-0_38`.

20 László Lovász. *Graphs and Geometry*, volume 65 of *Colloquium Publications*. AMS, Providence, RI, 2019.

21 Arturo I. Merino and Torsten Mütze. Combinatorial generation via permutation languages. III. rectangulations. *Discret. Comput. Geom.*, 70(1):51–122, 2023. `doi:10.1007/S00454-022-00393-W`.

22 William J. Mitchell, J. Philip Steadman, and Robin S. Liggett. Synthesis and optimization of small rectangular floor plans. *Environ. Plann. B Plann. Des.*, 3(1):37–70, 1976. `doi:10.1068/b030037`.

23 Takao Nishizeki and Md. Saidur Rahman. Rectangular drawing algorithms. In *Handbook on Graph Drawing and Visualization*, pages 317–348. Chapman and Hall/CRC, 2013.

24 Ralph H. J. M. Otten. Automatic floorplan design. In *Proc. 19th Design Automation Conference (DAC)*, pages 261–267. ACM/IEEE, 1982. `doi:10.1145/800263.809216`.

25 Md. Saidur Rahman, Shin-Ichi Nakano, and Takao Nishizeki. Rectangular grid drawings of plane graphs. *Comput. Geom.*, 10(3):203–220, 1998. `doi:10.1016/S0925-7721(98)00003-0`.

26 Md. Saidur Rahman, Shin-Ichi Nakano, and Takao Nishizeki. Rectangular drawings of plane graphs without designated corners. *Comput. Geom.*, 21(3):121–138, 2002. `doi:10.1016/S0925-7721(01)00061-X`.

27 Erwin Raisz. The rectangular statistical cartogram. *Geogr. Review*, 24(2):292–296, 1934. `doi:10.2307/208794`.

28 Nathan Reading. Generic rectangulations. *Eur. J. Comb.*, 33(4):610–623, 2012. `doi:10.1016/j.ejc.2011.11.004`.

29 Ingrid Rinsma. *Existence theorems for floorplans.* PhD thesis, University of Canterbury, Christchurch, New Zealand, 1987.

30 Ingrid Rinsma. Nonexistence of a certain rectangular floorplan with specified areas and adjacency. *Environ. Plann. B Plann. Des.*, 14(2):163–166, 1987. `doi:10.1068/b140163`.

31 Oded Schramm. Square tilings with prescribed combinatorics. *Israel J. Math.*, 84:97–118, 1993. `doi:10.1007/BF02761693`.

32 Carsten Thomassen. Plane representations of graphs. In *Progress in Graph Theory*, pages 43–69. Academic Press Canada, 1984.

33 William T. Tutte. Squaring the square. *Scientific American*, 199:136–142, 1958. In Gardner's 'Mathematical Games' column. Reprinted with addendum and bibliography in Martin Gardner, *The 2nd Scientific American Book of Mathematical Puzzles & Diversions*, pp. 186–209, Simon and Schuster, New York, 1961.

34 Peter Ungar. On diagrams representing maps. *J. London Math. Soc.*, s1-28(3):336–342, 1953. `doi:10.1112/jlms/s1-28.3.336`.

35 Kevin Weiler. Edge-based data structures for solid modeling in a curved surface environment. *IEEE Comput. Graph. Appl.*, 5(1):21–40, 1985. `doi:10.1109/MCG.1985.276271`.

36 Shmuel Wimer, Israel Koren, and Israel Cederbaum. Floorplans, planar graphs, and layouts. *IEEE Trans. Circuits Syst.*, 35(3):267–278, 1988. `doi:10.1109/31.1739`.

37 Bo Yao, Hongyu Chen, Chung-Kuan Cheng, and Ronald L. Graham. Floorplan representations: Complexity and connections. *ACM Trans. Design Autom. Electr. Syst.*, 8(1):55–80, 2003. `doi:10.1145/606603.606607`.

38 Gary K. H. Yeap and Majid Sarrafzadeh. Sliceable floorplanning by graph dualization. *SIAM J. Discret. Math.*, 8(2):258–280, 1995. `doi:10.1137/S0895480191266700`.