


RAC Drawings of Graphs with Low Degree

Patrizio Angelini ✉ 

John Cabot University, Rome, Italy

Michael A. Bekos ✉ 


Department of Mathematics, University of Ioannina, Ioannina, Greece

Julia Katheder ✉ 

Wilhelm-Schickard-Institut für Informatik, Universität Tübingen, Tübingen, Germany

Michael Kaufmann ✉ 

Wilhelm-Schickard-Institut für Informatik, Universität Tübingen, Tübingen, Germany

Maximilian Pfister ✉ 

Wilhelm-Schickard-Institut für Informatik, Universität Tübingen, Tübingen, Germany

Abstract

Motivated by cognitive experiments providing evidence that large crossing-angles do not impair the readability of a graph drawing, RAC (Right Angle Crossing) drawings were introduced to address the problem of producing readable representations of non-planar graphs by supporting the optimal case in which all crossings form 90° angles.

In this work, we make progress on the problem of finding RAC drawings of graphs of low degree. In this context, a long-standing open question asks whether all degree-3 graphs admit straight-line RAC drawings. This question has been positively answered for the Hamiltonian degree-3 graphs. We improve on this result by extending to the class of 3-edge-colorable degree-3 graphs. When each edge is allowed to have one bend, we prove that degree-4 graphs admit such RAC drawings, a result which was previously known only for degree-3 graphs. Finally, we show that 7-edge-colorable degree-7 graphs admit RAC drawings with two bends per edge. This improves over the previous result on degree-6 graphs.

2012 ACM Subject Classification Theory of computation → Computational geometry; Mathematics of computing → Graph algorithms

Keywords and phrases Graph Drawing, RAC graphs, Straight-line and bent drawings

Digital Object Identifier 10.4230/LIPIcs.MFCS.2022.11

Related Version *Full Version:* <https://arxiv.org/abs/2206.14909>

1 Introduction

In the literature, there is a wealth of approaches to draw planar graphs. Early results date back to Fáry's theorem [22], which guarantees the existence of a planar straight-line drawing for every planar graph; see also [9, 30, 31, 33, 34]. Over the years, several breakthrough results have been proposed, e.g., de Fraysseix, Pach and Pollack [11] in the late 80's devised a linear-time algorithm [10] that additionally guarantees the obtained drawings to be on an integer grid of quadratic size (thus making high-precision arithmetics of previous approaches unnecessary). Planar graph drawings have also been extensively studied in the presence of bends. Here, a fundamental result is by Tamassia [32] in the context of *orthogonal* graph drawings, i.e., drawings in which edges are axis-aligned polylines. In his seminal paper, Tamassia suggested an approach, called *topology-shape-metrics*, to minimize the number of bends of degree-4 plane graphs using flows. For a complete introduction, see [12].

When the input graph is non-planar, however, the available approaches that yield aesthetically pleasing drawings are significantly fewer. The main obstacle here is that the presence of edge-crossings negatively affects the drawing's quality [28] and, on the other hand,



© Patrizio Angelini, Michael A. Bekos, Julia Katheder, Michael Kaufmann, and Maximilian Pfister; licensed under Creative Commons License CC-BY 4.0

47th International Symposium on Mathematical Foundations of Computer Science (MFCS 2022).

Editors: Stefan Szeider, Robert Ganian, and Alexandra Silva; Article No. 11; pp. 11:1–11:15

Leibniz International Proceedings in Informatics



LIPICs Schloss Dagstuhl – Leibniz-Zentrum für Informatik, Dagstuhl Publishing, Germany

their minimization turns out to be a computationally difficult problem [23]. In an attempt to overcome these issues, a decade ago, Huang et al. [26] made a crucial observation that gave rise to a new line of research (currently recognized under the term “beyond planarity” [25]): edge crossings do not negatively affect the quality of the drawing too much (and hence the human’s ability to read and interpret it), if the angles formed at the crossing points are large. Thus, the focus moved naturally to non-planar graphs and their properties, when different restrictions on the type of edge-crossings are imposed; see [19] for an overview.

Among the many different classes of graphs studied as part of this emerging line of research, one of the most studied ones is the class of *right-angle-crossing graphs* (or *RAC graphs*, for short); see [14] for a survey. These graphs were introduced by Didimo, Eades and Liotta [15, 17] back in 2009 as those admitting straight-line drawings in which the angles formed at the crossings are all 90° . Most notably, these graphs are optimal in terms of the crossing angles, which makes them more readable according to the observation by Huang et al. [26]; moreover, RAC drawings form a natural generalization of orthogonal graph drawings [32], as any crossing between two axis-aligned polylines trivially yields 90° angles.

In the same work [15, 17], Didimo, Eades and Liotta proved that every n -vertex RAC graph is sparse, as it can contain at most $4n - 10$ edges, while in a follow-up work [16] they observed that not all degree-4 graphs are RAC. This gives rise to the following question which has also been independently posed in several subsequent works (see e.g., [3], [18, Problem 6], [14, Problem 9.5], [19, Problem 8]) and arguably forms the most intriguing open problem in the area, as it remains unanswered since more than one decade.

► **Question 1.** *Does every graph with degree at most 3 admit a straight-line RAC drawing?*

The most relevant result that is known stems from the related problem of simultaneously embedding two or more graphs on the Euclidean plane, such that the crossings between different graphs form 90° angles. In this setting, Argyriou et al. [4] showed that a cycle and a matching always admit such an embedding, which implies that every *Hamiltonian* degree-3 graph is RAC.

Finally, note that recognizing RAC graphs is hard in the existential theory of the reals [29], which also implies that RAC drawings may require double-exponential area, in contrast to the quadratic area requirement for planar graphs [11].

RAC graphs have also been studied by relaxing the requirement that the edges are straight-line segments, giving rise to the class of *k-bend RAC graphs* (see, e.g. [1, 6, 7, 8, 13]), i.e., those admitting drawings with at most k bends per edge and crossings at 90° angles. It is known that every degree-3 graph is 1-bend RAC and every degree-6 graph is 2-bend RAC [3]. While the flexibility guaranteed by the presence of one or two bends on each edge is not enough to obtain a RAC drawing for every graph (in fact, 1- and 2-bend RAC graphs with n vertices have at most $5.5n - O(1)$ and $72n - O(1)$ edges, respectively [1, 6]), it is known that every graph is 3-bend RAC [13] and fits on a grid of cubic size [21].

Our contribution. We provide several improvements to the state of the art concerning RAC graphs with low degree. In particular, we make an important step towards answering Question 1 by proving that 3-edge-colorable degree-3 graphs are RAC (Theorem 3). This result applies to Hamiltonian 3-regular graphs, to bipartite 3-regular graphs and, with some minor modifications to our approach, to all Hamiltonian degree-3 graphs, thus extending the result in [4]. As a further step towards answering Question 1, we prove that bridgeless 3-regular graphs with oddness at most 2 are RAC (Theorem 8). If their oddness is k , we provide an algorithm to construct a 1-bend RAC drawing where at most k edges have a bend (Theorem 9).

We then focus on RAC drawings with one or two bends per edge. Namely, we prove that all degree-4 graphs admit 1-bend RAC drawings and all 7-edge-colorable degree-7 graphs admit 2-bend RAC drawings (Theorems 10 and 12), which form non-trivial improvements over the state of the art, as the existence of such drawings was previously known only for degree-3 and degree-6 graphs [3]. Due to space constraints, proofs of statements marked with (*) can be found in [2].

2 Preliminaries

Let $G = (V, E)$ be a graph. W.l.o.g. we assume that G is connected, as otherwise we apply our drawing algorithms to each component of G separately. G is called *degree- k* if the maximum degree of G is k . G is called *k -regular* if the degree of each vertex of G is exactly k . A 2-factor of an undirected graph $G = (V, E)$ is a spanning subgraph of G consisting of vertex disjoint cycles. Let F be a 2-factor of G and let \prec be a total order of the vertices such that the vertices of each cycle $C \in F$ appear consecutive in \prec according to some traversal of C . In other words, every two vertices that are adjacent in C are consecutive in \prec except for two particular vertices, which are the first and the last vertices of C in \prec . We call the edge between these two vertices the *closing edge* of C . By definition, \prec also induces a total order of the cycles of F . Let $\{u, v\}$ be an edge in $E \setminus F$ and let C and C' be the cycles of F that contain u and v , respectively. If $C = C'$, then $\{u, v\}$ is a *chord* of C . Otherwise, $C \neq C'$. If $u \prec v$, $\{u, v\}$ is called a *forward edge* of u and a *backward edge* of v . The following theorem provides a tool to partition the edges of a bounded degree graph into 2-factors [27].

► **Theorem 2** (Eades, Symvonis, Whitesides [20]). *Let $G = (V, E)$ be an n -vertex undirected graph of degree Δ and let $d = \lceil \Delta/2 \rceil$. Then, there exists a directed multi-graph $G' = (V, E')$ with the following properties:*

1. *each vertex of G' has indegree d and outdegree d ;*
2. *G is a subgraph of the underlying undirected graph of G' ; and*
3. *the edges of G' can be partitioned into d edge-disjoint directed 2-factors.*

Furthermore, the directed graph G' and the d 2-factors can be computed in $\mathcal{O}(\Delta^2 n)$ time.

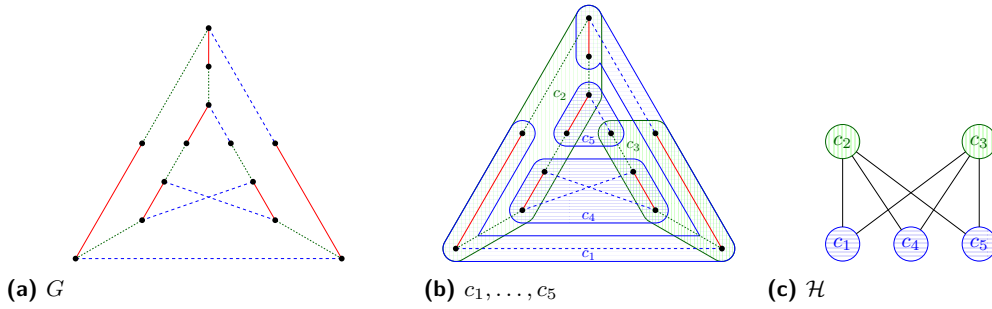
Let Γ be a polyline drawing of G such that the vertices and edge-bends lie on grid points. The area of Γ is determined by the smallest enclosing rectangle. Let $\{u, v\}$ be an edge in Γ . We say that $\{u, v\}$ is using an *orthogonal port* at u if the edge-segment s_u of $\{u, v\}$ that is incident to u is either horizontal or vertical; otherwise it is using an *oblique port* at u . We denote the orthogonal ports at u by N , E , S and W , if s_u is above, to the right, below or to the left of u , respectively. If no edge is using a specific orthogonal port, we say that this port is *free*.

3 RAC drawings of 3-edge-colorable degree-3 graphs

In this section, we prove that 3-edge-colorable degree-3 graphs admit RAC drawings of quadratic area, which can be computed in linear time assuming that the edge coloring is given (testing the existence of such a coloring is NP-complete even for 3-regular graphs [24]).

► **Theorem 3.** *Given a 3-edge-colorable degree-3 graph G with n vertices and a 3-edge-coloring of G , it is possible to compute in $\mathcal{O}(n)$ time a RAC drawing of G with $\mathcal{O}(n^2)$ area.*

We assume w.l.o.g. that G does not contain degree-1 vertices, as otherwise we can replace each such vertex with a 3-cycle while maintaining the 3-edge-colorability of the graph and without asymptotically increasing the size of the graph. Since G is 3-edge-colorable, it can be



■ **Figure 1** (a) A non-planar, non-Hamiltonian, 3-edge-colorable degree-3 example graph G . The matching M_1 is drawn with blue dashed lines, M_2 with red solid lines and M_3 with green dotted lines. (b) The components c_1, c_2 and c_3 of the subgraph H_y induced by $M_1 \cup M_2$ (shaded in blue) and the components c_4 and c_5 of H_x induced by $M_2 \cup M_3$ (shaded in green). (c) The auxiliary graph \mathcal{H} in which the components of H_y and H_x sharing at least one vertex are connected by an edge. In this example, the BFS traversal of the components of \mathcal{H} is c_1, c_2, c_3, c_4, c_5 .

decomposed into three matchings M_1, M_2 and M_3 . In the produced RAC drawing, the edges in M_1 will be drawn horizontal, those in M_3 vertical, while those in M_2 will be crossing-free, not maintaining a particular slope. Let H_y and H_x be two subgraphs of G induced by $M_1 \cup M_2$ and $M_2 \cup M_3$, respectively. Since every vertex of G has at least two incident edges, which belong to different matchings, each of H_y and H_x spans all vertices of G . Further, any connected component in H_y or H_x is either a path or an even-length cycle, as both H_y and H_x are degree-2 graphs alternating between edges of different matchings.

We define an auxiliary bipartite graph \mathcal{H} , whose first (second) part has a vertex for each connected component in H_y (H_x), and there is an edge between two vertices if and only if the corresponding components share at least one vertex; see Fig. 1.

► **Property 4.** *The auxiliary graph \mathcal{H} is connected.*

Proof. Suppose for a contradiction that \mathcal{H} is not connected. Let v_c and $v_{c'}$ be two vertices of \mathcal{H} that are in different connected components of \mathcal{H} . By definition of \mathcal{H} , v_c and $v_{c'}$ correspond to connected components c and c' , respectively, of H_y or H_x . W.l.o.g. assume that c belongs to H_y . Let u_0 and u_k be two vertices of G that belong to c and c' , respectively. Since G is connected, there is a path $P = (u_0, u_1, u_2, \dots, u_k)$ between u_0 and u_k in G , such that no two consecutive edges in P belong to the same matching. Let (u_i, u_{i+1}) be the first edge of P from u_0 to u_k that belongs to M_3 , which exists since $c \neq c'$. By construction, this implies that u_i belongs to c and to another component c^* of H_x . By definition of \mathcal{H} , v_c and v_{c^*} are connected in \mathcal{H} , where v_{c^*} is the corresponding vertex of c^* in \mathcal{H} . Repeating this argument until u_k is reached yields a path in \mathcal{H} from v_c to $v_{c'}$, a contradiction. ◀

We now define two total orders \prec_y and \prec_x of the vertices of G , which will then be used to assign their y - and x -coordinates, respectively, in the final RAC drawing of G . Since we seek to draw the edges of M_1 (M_3) horizontal (vertical), we require that the endvertices of any edge in M_1 (M_3) are consecutive in \prec_y (\prec_x , respectively). Moreover, for the edges of M_2 , we guarantee some properties that will allow us to draw them without crossings.

To construct \prec_y and \prec_x , we process the components of H_y and H_x according to a certain BFS traversal of \mathcal{H} and for each visited component of H_y (H_x), we append all its vertices to \prec_y (\prec_x) in a certain order.

To select the first vertex of the BFS traversal of \mathcal{H} , we consider a vertex u of G belonging to two components c and c' of H_y and H_x , respectively, such that u is the endpoint of c if c is a path; if c is a cycle, we do not impose any constraints on the choice of u . We refer to

vertex u as the *origin vertex* of G . Also, let v_c and $v_{c'}$ be the vertices of \mathcal{H} corresponding to c and c' , respectively. By definition of \mathcal{H} , v_c and $v_{c'}$ are adjacent in \mathcal{H} . We start our BFS traversal of \mathcal{H} at v_c and then we move to $v_{c'}$ in the second step (note that this choice is not needed for the definition of \prec_y and \prec_x , but it guarantees a structural property that will be useful later). From this point on, we continue the BFS traversal to the remaining vertices of \mathcal{H} without further restrictions. In the following, we describe how to process the components of H_y and H_x in order to guarantee an important property (see Property 5)

Let c be the component of H_y or H_x corresponding to the currently visited vertex in the traversal of \mathcal{H} . Since \mathcal{H} is bipartite, no other component of H_y (H_x) shares a vertex with c , if c belongs to H_y (H_x). Hence, no vertex of c already appears in \prec_y (\prec_x).

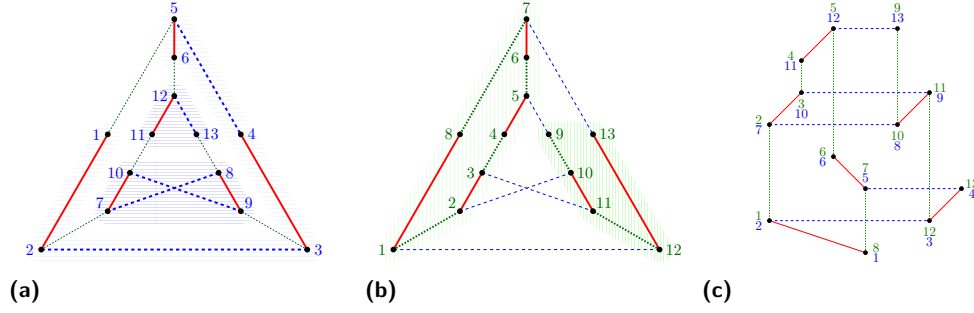
If c is a path, then we append the vertices of c to \prec_y or \prec_x in the natural order defined by a walk from one of its endvertices to the other. Note that if c is the first component in the BFS traversal of \mathcal{H} , one of these endvertices is by definition the origin vertex of G , which we choose to start the walk. Hence, in the following we focus on the case that c is a cycle. In this case, the vertices of c will also be appended to \prec_y or \prec_x in the natural order defined by some specific walk of c , such that the so-called *closing edge* connecting the first and the last vertex of c in this order belongs to M_2 . Note that an edge might be closing in both orders \prec_y and \prec_x .

Suppose first that $c \in H_y$. If c is the first component in the BFS traversal of \mathcal{H} , then we append the vertices of c to \prec_y in the order that they appear in the cyclic walk of c starting from the origin vertex of G and following the edge of M_1 incident to it. Otherwise, let v be the first vertex of c in \prec_x , which is well defined since there is at least one vertex of c that is part of \prec_x , namely, the one that is shared with its parent. We append the vertices of c to \prec_y in the order that they appear in the cyclic walk of c starting from v and following the edge of M_1 incident to v . Hence, v is the first vertex of c in both \prec_y and \prec_x . In both cases, it follows that the closing edge of c belongs to M_2 .

Suppose now that $c \in H_x$, which implies that c is not the first component in the BFS traversal of \mathcal{H} . Let v be the first vertex of c in \prec_y , which is again well defined since there is at least one vertex of c that is part of \prec_y . We append the vertices of c to \prec_x in the inverse order that they appear in the cyclic walk of c starting from v and following the edge (v, w) of M_3 incident to v (or equivalently, in the order they appear in the cyclic walk of c starting from the neighbor of v different from w and ending at v). Hence, v is the first vertex of c in \prec_y and the last vertex of c in \prec_x . Also in this case, the closing edge of c belongs to M_2 . See Fig. 2 for an illustration. Note that the closing edge of a component c is contained inside the parent component of c is the BFS traversal. Moreover, by construction, the following property holds.

► **Property 5.** *The endvertices of any edge in M_1 (M_3) are consecutive in \prec_y (\prec_x). The endvertices of any edge in M_2 are consecutive in \prec_y (\prec_x) unless this edge is a closing edge in a component of H_y (of H_x).*

Computing the vertex coordinates. We use \prec_y and \prec_x to specify the y - and the x -coordinates of the vertices, respectively. To do so, we iterate through \prec_y and set the y -coordinate of its first vertex to 1. Let v be the next vertex in the iteration and let u be its predecessor in \prec_y . Assume that the y -coordinate of u is i . If $(u, v) \in M_1$, we set the same y -coordinate i to v . Otherwise, either u and v belong to two different components of H_y or $(u, v) \in M_2$ and we set the y -coordinate $i + 1$ to v . Similarly, we iterate through \prec_x and set the x -coordinate of its first vertex to 1. Let v be the next vertex in the iteration and let u be its predecessor in \prec_x . Assume that the x -coordinate of u is i . If $(u, v) \in M_3$, we set



■ **Figure 2** The total orders (a) \prec_y for H_y that consists of the blue and red edges, and (b) \prec_x for H_x that consists of the green and red edges. The final drawing of G is shown in (c).

the x -coordinate of v to i . Otherwise, either u and v belong to two different components of H_x or $(u, v) \in M_2$ and we set the x -coordinate $i + 1$ to v . Hence, no two vertices share the same x - and y -coordinates. We next show that the computed vertex coordinates induce a straight-line RAC drawing Γ of G with the possible exception of the edge of M_2 incident to the origin vertex of G , since this edge would be a closing edge for both c and c' and hence by Property 5 its endpoints would be consecutive in neither \prec_y nor \prec_x . If this edge exists, we denote it by e^* , while the graph $G \setminus e^*$ and its drawing in Γ are denoted by G^* and Γ^* , respectively.

► **Lemma 6.** *Let e be an edge of G^* . Then, e is drawn horizontally in Γ^* if $e \in M_1$; vertically in Γ^* if $e \in M_3$ and crossing-free in Γ^* if $e \in M_2$.*

Proof. If $e \in M_1$ or $e \in M_3$, the statement follows from Property 5 and the computed vertex coordinates. Hence, let $e = (u, v)$ be an edge of M_2 and let $c_y \in H_y$ and $c_x \in H_x$ be the two components of \mathcal{H} containing e . Suppose to the contrary that there is an edge $e' = (u', v')$ crossing e . If $e' \in M_1$, then both u' and v' belong to the same component $c'_y \in H_y$. If $c'_y \neq c_y$, then e' cannot cross e as the vertices of c_y and c'_y span different intervals of y -coordinates, hence e' belongs to c_y . Similarly, if $e' \in M_3$, then e' belongs to c_x . Finally, if $e' \in M_2$, then u' and v' belong to the same component in both H_y and H_x ; thus e' belongs to both c_y and c_x .

1. Edge e is a closing edge for neither c_y nor c_x : The vertices u and v are consecutive in both \prec_y and \prec_x by Property 5. Hence, both their x - and y -coordinate differ by exactly one by construction. Since all vertices have integer coordinates, no horizontal or vertical edge is crossing e , hence $e' \in M_2$. Observe that since the y - (the x -) coordinate of the vertices in c_y (in c_x) are non-decreasing along the walk defining its order in \prec_y (in \prec_x), no crossing between e and e' can occur if e' is not a closing edge of c_y or c_x , which will be covered in the next cases (by swapping the roles of e and e').
2. Edge e is a closing edge for c_y but not for c_x : Note that c_y is not the first component in the BFS traversal, since the closing edge of this component is e^* . Further, u and v are consecutive in \prec_x , but not in \prec_y . We assume that u directly precedes v in \prec_x , which implies that u is the first vertex of c_y in \prec_y , while v is the last. It follows that their x -coordinates differ by exactly one, hence e' cannot belong to M_3 . If e' belongs to M_1 , then one of u' or v' , say u' , has x -coordinate smaller or equal to the one of u . Since u and v are consecutive in \prec_x , we have necessarily that u' precedes u in \prec_x , which is a contradiction to the choice of u since both u' and v' belong to c_y . In fact, u was chosen as the starting point of the walk, when considering c_y , as the first vertex of c_y in \prec_y , hence $e' \in M_2$. Since both endpoints of e' belong to both c_y and c_x , then one of u' or v' ,

say u' , has x -coordinate smaller or equal to the one of u . Since u and v are consecutive in \prec_x , we have necessarily that $u' \prec_x u$, which is a contradiction to the choice of u since both u' and v' belong to c_y .

3. Edge e is a closing edge for c_x but not for c_y : This case is analogous to the previous one.
4. Edge e is a closing edge for both c_y and c_x : Observe that neither c_y nor c_x is the first component in the BFS traversal of \mathcal{H} , since e^* is the closing edge of this component, which is not part of G^* . Recall that by definition, the vertices v_{c_y} and v_{c_x} corresponding to c_y and c_x in \mathcal{H} are adjacent. Assume that c_y is visited before c_x in the BFS traversal; the other case is symmetric. By our construction rule, when considering c_y , we started the walk from the vertex u that is the first vertex of c_y in \prec_x , which means u also belongs to a component c'_x of H_x . Since c_y is a cycle, u is incident to an edge in M_2 , which then also belongs to c'_x . Clearly, the edge of M_2 incident to u is the closing edge of c_y and contained in c'_x , which implies that the edge does not belong to c_x , hence this case does not occur in G^* . ◀

By the last case of Lemma 6, it follows that if the edge e^* exists, then it is the only closing edge of two components, which is summarized in the following corollary.

► **Corollary 7.** *There is at most one edge in M_2 that is a closing edge for two components.*

We now describe how to add the edge e^* to Γ^* if such an edge exists to obtain the final drawing Γ . Let u and v be the endvertices of e^* with u being the origin vertex of G . By construction, u and v are in the first two components c and c' of the BFS traversal of \mathcal{H} . Since u is the first vertex in \prec_y , its y -coordinate is 1, i.e., u is the bottommost vertex of Γ^* . Also, since u is the first vertex of c' in \prec_y , it is incident to the closing edge of c' and c by definition, in particular, this edge is e^* . Note that this implies that the x -coordinate of v is 1, so v is the leftmost vertex of Γ^* and the first vertex in \prec_x . This ensures that v can be moved to the left and u to the bottom in order to draw the edge e^* crossing-free. In particular, moving v by n units to the left and u by n units to the bottom we can guarantee that e^* does not intersect the first quadrant \mathbb{R}_+^2 , while by construction any other edge (not incident to u or v) lies in \mathbb{R}_+^2 . Since e^* is the only edge of M_2 incident to u and v , it remains to consider the edges of M_1 and M_3 incident to u or v . Observe that the edge of M_1 incident to v remains horizontal, while the edge of M_3 incident to u remains vertical. Finally, the edge of M_1 incident to u is crossing free in Γ^* , since there is no vertex below it, hence it remains crossing-free after moving u to the bottom. Similarly, the edge of M_3 incident to v is crossing free in Γ^* , since there is not vertex to the left of it, hence it remains crossing-free after moving v to the left. Together with Lemma 6 we obtain that Γ is a RAC drawing of G . We complete the proof of Theorem 3 by discussing the time complexity and the required area. We construct the components of H_y and H_x based on the given edges-coloring using BFS in $\mathcal{O}(n)$ time. To define \prec_y and \prec_x , we choose the origin vertex u and the components c and c' for the start of the BFS of \mathcal{H} in linear time. We then traverse every edge of G at most twice. Hence, this step takes $\mathcal{O}(n)$ time in total. Assigning the vertex coordinates, by first iterating through \prec_y and \prec_x and then possibly moving the end-vertices of e^* , clearly takes $\mathcal{O}(n)$ time again, hence the drawing can be computed in linear time.

For the area, we observe that the initial x - and y -coordinates for all the vertices range between 1 and n . Since we possibly move the origin vertex u and its M_2 neighbor v by n units each, the drawing area is at most $2n \times 2n$.

We conclude this section by mentioning two results that form generalizations of our approach of Theorem 3. In this regard, we need the notion of oddness of a bridgeless 3-regular graph, which is defined as the minimum number of odd cycles in any possible 2-factor of it. Theorem 8 is limited to oddness-2, while Theorem 9 provides an upper bound on the number of edges requiring one bend that is linear in the oddness; their proofs are in [2].

► **Theorem 8. (*)** *Every bridgeless 3-regular graph with oddness 2 admits a RAC drawing in quadratic area which can be computed in subquadratic time.*

► **Theorem 9. (*)** *Every bridgeless 3-regular graph with oddness $k \geq 2$ admits a 1-bend RAC drawing in quadratic area where at most k edges require one bend.*

4 1-Bend RAC Drawings of Degree-4 graphs

In this section, we focus on degree-4 graphs and show that they admit 1-bend RAC drawings.

► **Theorem 10.** *Given a degree-4 graph G with n vertices, it is possible to compute in $O(n)$ time a 1-bend RAC drawing of G with $O(n^2)$ area.*

Proof. By Theorem 2, we augment G into a directed 4-regular multigraph G' with edge disjoint 2-factors F_1 and F_2 . Let G_s be the graph obtained from G' as follows. For each vertex u of G' with incident edges $(a_1, u), (u, b_1) \in F_1$ and $(a_2, u), (u, b_2) \in F_2$, we add two vertices u_s and u_t to G_s that are incident to the following five edges: u_s is incident to the two incoming edges of u , namely, (a_1, u_s) and (a_2, u_s) , while u_t is incident to (u_t, b_1) and (u_t, b_2) . Finally, we add the edge (u_s, u_t) to G_s , which we call *split-edge*.

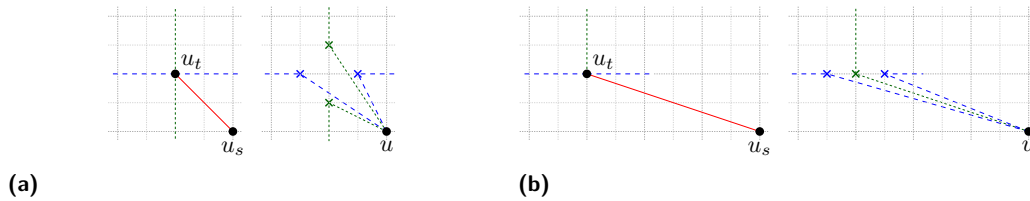
By construction, G_s is 3-regular and 3-edge colorable, since each vertex of it is incident to one edge of F_1 , one edge of F_2 and one split-edge. By applying the algorithm of Theorem 3 to G_s , we obtain a RAC drawing Γ_s of G_s , such that the matching M_2 in the algorithm is the one consisting of all the split-edges. To obtain a 1-bend drawing for G' , it remains to merge the vertices u_s and u_t for every vertex u in G' . We place u at the position of u_s in Γ_s . We draw each outgoing edge (u, x) of u as a polyline with a bend placed close to the position of u_t in Γ_s (the specific position will be discussed later), which implies that the two segments are close to the edges (u_s, u_t) and (u_t, x) in Γ_s , respectively. This guarantees that any edge has exactly one bend, as any edge is outgoing for exactly one of its endvertices.

We next discuss how we place the bends for the outgoing edges of u . Since each split-edge belongs to M_2 , it is drawn in Γ_s either as the diagonal of a 1×1 grid box or as a closing edge. Also, since the outgoing edges of u are in M_1 and M_3 , they are drawn as horizontal and vertical line-segments.

Assume first that the split-edge of u is the diagonal of a 1×1 grid box; see Fig. 3a. If the outgoing edge (u_t, x) belongs to M_1 , then we place the bend of (u, x) either half a unit to the right of u_t if x is to the right of u_t in Γ_s , or half a unit to its left otherwise. Symmetrically, if (u_t, x) belongs to M_3 , we place the bend half a unit either above or below u_t .

Assume now that the split-edge of u is a closing edge in exactly one of H_y or H_x , say w.l.o.g. of a cycle c in H_x , i.e., it spans the whole x -interval of c . By construction, the outgoing edge of (u_t, x) that belongs to M_3 is a vertical line-segment attached above u , as either u_t or u_s are the first vertex of c in \prec_y ; in the latter case, u_t is the second vertex of c in \prec_y by construction. If the edge (u_t, x) belongs to M_3 , we place the bend exactly at the computed position of u_t . If (u_t, x) belongs to M_1 , we place it either half a unit to the right of u_t if x is to the right of u_t in Γ_s , or half a unit to its left otherwise; see Fig. 3b.

Assume last that the split-edge of u is the closing edge e for a H_y and a H_x cycle, which is unique by Corollary 7. As discussed for the analogous case in Section 3 (see the discussion following Corollary 7), one of u_s and u_t , say w.l.o.g. u_s , is the leftmost, while the other u_t is the bottommost vertex in Γ_s . For the placement of the bends, we slightly deviate from our approach above. Let (u_t, x) and (u_t, y) be the two edges of M_1 and M_3 incident to u_t in G_s . Then, it is not difficult to find two grid points b_x and b_y sufficiently below the positions of x and y in Γ_s , such that (u, x) and (u, y)



■ **Figure 3** Illustration on how to place the bends in the proof of Theorem 10. To merge the vertices u_s and u_t of a vertex u in G' , u is placed at the position of u_s . The bends of the outgoing edges at u are placed close to the position of u_t in the drawing depending on their orientation.

drawn by bending at b_x and b_y do not cross. Since no two bends overlap, no new crossings are introduced and the slopes of the segments involved in crossings are not modified, the obtained drawing Γ' is a 1-bend RAC drawing for G' (and thus for G).

Regarding the time complexity, we observe that we can apply Theorem 2 and the split-operation in $\mathcal{O}(n)$ time. The split operation immediately yields a valid 3-coloring of the edges, hence we can apply the algorithm of Theorem 3 to obtain Γ_s in $\mathcal{O}(n)$ time. Finally, contracting the edges can clearly be done in $\mathcal{O}(n)$ time, as it requires a constant number of operations per edge. For the area, we observe that in order to place the bends, we have to introduce new grid-points, but we at most double the number of points in any dimension, hence we still maintain the asymptotic quadratic area guaranteed by Theorem 3. ◀

The following theorem, whose proof is in [2], provides an alternative construction which additionally guarantees a linear number of edges drawn as straight-line segments.

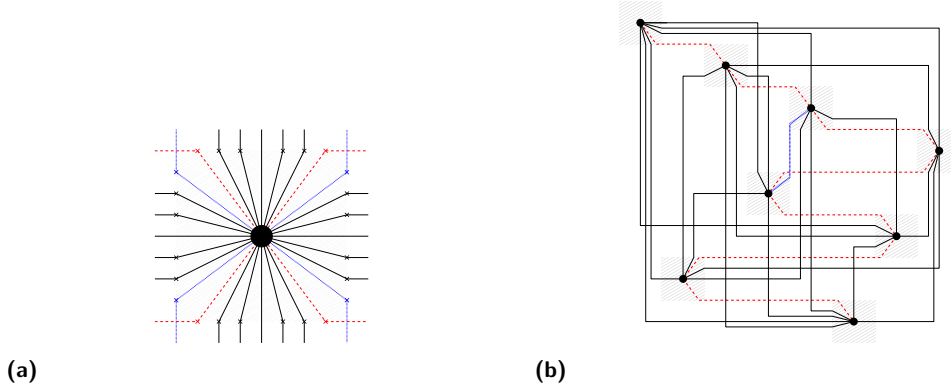
► **Theorem 11. (*)** *Given a degree-4 graph G with n vertices and m edges, it is possible to compute in $\mathcal{O}(n)$ time a 1-bend RAC drawing of G with $\mathcal{O}(n^2)$ area where at least $\frac{m}{8}$ edges are drawn as straight-line segments.*

5 RAC drawings of 7-edge-colorable degree-7 graphs

We prove that 7-edge-colorable degree-7 graphs admit 2-bend RAC drawings by proving the following slightly stronger statement.

► **Theorem 12.** *Given a degree-7 graph G decomposed into a degree-6 graph H and a matching M , it is possible to compute in $\mathcal{O}(n)$ time a 2-bend RAC drawing of G with $\mathcal{O}(n^2)$ area.*

Since H is a degree-6 graph, it admits a decomposition into three disjoint (directed) 2-factors F_1 , F_2 and F_3 after applying Theorem 2 and (possibly) augmenting H to a 6-regular (multi)-graph. To distinguish between directed and undirected edges, we write $\{u, v\}$ to denote an undirected edge between u and v , while (u, v) denotes a directed edge from u to v . In the following, we will define two total orders \prec_x and \prec_y , which will define the x - and y -coordinates of the vertices of G , respectively. We define \prec_y such that the vertices of each cycle in F_1 will be consecutive in \prec_y . Initially, for any cycle of F_1 , the specific internal order of its vertices in \prec_y is specified by one of the two traversals of it; however, we note here that this choice may be refined later in order to guarantee an additional property (described in Lemma 13). The definition of \prec_x is more involved and will also be discussed later. Theorem 2 guarantees that the edges of F_2 (F_3) are oriented such that any vertex has at most one incoming and one outgoing edge in F_2 (F_3). Once \prec_y and \prec_x are computed, each vertex u of G will be mapped to point $(8i, 8j)$ of the Euclidean plane provided that u is the i -th vertex in \prec_x and the j -th vertex in \prec_y . Each vertex u is associated with a closed box $B(u)$ centered at u of size 8×8 . We aim at computing a drawing of G in which



■ **Figure 4** Edge routing in the 8×8 box $B(u)$ of a vertex u (a). Red dashed (blue dotted) ports are reserved for horizontal (vertical) type-2 edges. A 2-bend RAC drawing of K_8 is shown in (b). In this drawing, red dashed edges are horizontal type-2, while the blue dotted one is vertical type-2.

- (i) no two boxes overlap, and
- (ii) the edges are drawn with two bends each so that only the edge-segments that are incident to u are contained in the interior of $B(u)$, while all the other edge-segments are either vertical or horizontal.

This guarantees that the resulting drawing is 2-bend RAC; see Fig. 4b.

In the final drawing, all edges will be drawn with exactly three segments, out of which either one or two are *oblique*, i.e., they are neither horizontal nor vertical. It follows from (ii) that the bend point between an oblique segment and a vertical (horizontal) segment lies on a horizontal (vertical) side of the box containing the oblique segment. During the algorithm, we will classify the edges as *type-1* or *type-2*. Type-1 edges will be drawn with one oblique segment, while type-2 edges with two oblique segments. In particular, for a type-1 edge (u, v) , we further have that the oblique segment is incident to v , which implies that (u, v) occupies an orthogonal port at u . On the other hand, a type-2 edge (u, v) requires that $B(u)$ and $B(v)$ are *aligned* in y (in x), i.e., there exists a horizontal (vertical) line that is partially contained in both $B(u)$ and $B(v)$, in order to draw the middle segment of (u, v) horizontally (vertically). By construction, this is equivalent to having u and v consecutive in \prec_y (\prec_x). These alignments guarantee that if we partition the edges of F_1 into \bar{F}_1 and \hat{F}_1 containing the closing and non-closing ones, respectively, then it is possible to draw \hat{F}_1 as a horizontal type-2 edge (independent of the x -coordinate of its endvertices), as its endvertices are consecutive in \prec_y by construction. Thus, we can put our focus on edges in $\bar{F}_1 \cup F_2 \cup F_3$, which we initially classify as type-1 edges (by orienting each edge (u, v) of \bar{F}_1 from u to v if $u \prec_y v$). We refine \prec_y using the concept of *critical vertices*. Namely, for a vertex u of G , the direct successors of u in $\bar{F}_1 \cup F_2 \cup F_3$ are the critical vertices of u , which are denoted by $c(u)$. Based on the relative position of u to its critical vertices in \prec_y , we label u as (α, β) , if α vertices t_1, \dots, t_α of $c(u)$ are after u in \prec_y and β vertices b_1, \dots, b_β before. We refer to t_1, \dots, t_α (b_1, \dots, b_β) as the *upper* (*lower*) *critical neighbors* of u . An edge connecting u to an upper (lower) critical neighbor is called *upper critical* (*lower critical*, respectively). More general, the upper and lower critical edges of u are its *critical edges*.

Note that $2 \leq \alpha + \beta \leq 3$ as any vertex has exactly one outgoing edge in F_2 and F_3 and at most one in \bar{F}_1 , that is, the number of upper and lower critical neighbors of vertex u ranges between 2 and 3. It follows that the label of each vertex of H is in $\{(0, 2), (1, 1), (2, 0), (1, 2), (2, 1), (3, 0)\}$; refer to these labels as the *feasible labels* of the vertex.

Observe that a $(3,0)$ -, $(2,1)$ - or $(1,2)$ -label implies that the vertex is incident to a closing edge of F_1 (hence, each cycle in F_1 has at most one such vertex, which is its first one in \prec_y). This step will complete the definition of \prec_y .

- **Lemma 13.** (*) *For each cycle c of F_1 , there is an internal ordering of its vertices followed by a possible reorientation of one edge in $F_2 \cup F_3$, such that in the resulting \prec_y*
- (a) *every vertex of c has a feasible label,*
 - (b) *no vertex of c has label $(3,0)$, and*
 - (c) *if there exists a $(1,2)$ -labeled vertex in c , then its (only) upper critical neighbor belongs to c .*

Now that \prec_y is completely defined, we orient any edge $(u, v) \in M$ from u to v if and only if $u \prec_y v$. In this case, we further add v as a critical vertex of u . This implies that some vertices can have one more critical upper neighbor, which then gives rise to the new following labels, which we call *tags* for distinguishing: $\{[3, 1], [3, 0], [2, 2], [2, 1], [2, 0], [1, 2], [1, 1], [0, 2]\}$. In this context, Lemma 13 guarantees the following property.

- **Property 14.** *Any cycle c of F_1 has at most one vertex with tag $[\alpha, \beta]$ such that $\alpha + \beta = 4$.*

Next, we compute the final drawing satisfying Properties (i) and (ii) by performing two iterations over the vertices of G in reverse \prec_y order. In the first, we specify the final position of each vertex of G in \prec_x and classify its incident edges while maintaining the following Invariant 15. In the second one, we exploit the computed \prec_x to draw all edges of G .

- **Invariant 15.** *The endvertices of each vertical type-2 edge are consecutive in \prec_x . Further, any vertex is incident to at most one vertical type-2 edge.*

The second part of Invariant 15 implies that the vertical type-2 edges form a set of independent edges. In this regard, we say that a vertex u is a *partner* of a vertex v in G if and only if u and v are connected with an edge in this set.

In the first iteration, we assume that we have processed the first i vertices v_n, \dots, v_{n-i+1} of G in reverse \prec_y order and we have added these vertices to \prec_x together with a classification of their incident edges satisfying Invariant 15. We determine the position of v_{n-i} in \prec_x based on the \prec_x position of its upper critical neighbors. The incident edges of v_{n-i} are classified based on a case analysis on its tag $[\alpha, \beta]$. Recall that unless otherwise specified, every edge is a type-1 edge.

1. The tag of v_{n-i} is $[3, 1]$ or $[3, 0]$: Let a , b and c be the upper critical neighbors of v_{n-i} , which implies that they were processed before v_{n-i} by the algorithm and are already part of \prec_x . W.l.o.g. assume that $a \prec_x b \prec_x c$. By Invariant 15, vertex b is the partner of at most one already processed vertex b' , which is consecutive with b in \prec_x . If b' exists and $b' \prec_x b$, then we add v_{n-i} immediately after b in \prec_x . Symmetrically, if b' exists and $b \prec_x b'$, then we add v_{n-i} immediately before b in \prec_x . Otherwise, we add v_{n-i} immediately before b in \prec_x . This guarantees that v_{n-i} is placed between a and c in \prec_x and that Invariant 15 is satisfied, since none of the upper critical edges incident to v_{n-i} was classified as a type-2 edge.
2. The tag of v_{n-i} is $[2, 1]$, $[2, 0]$, $[1, 2]$, $[1, 1]$ or $[0, 2]$: By appending v_{n-i} to \prec_x , we maintain Invariant 15, since none of the upper critical edges incident to v_{n-i} was classified as type-2.
3. The tag of v_{n-i} is $[2, 2]$: Let a and b be the upper critical neighbors of v_{n-i} , which implies that they were processed before v_{n-i} by the algorithm and are already part of \prec_x . W.l.o.g.

assume that $(v_{n-i}, a) \in M$. We classify the edge (v_{n-i}, b) as a vertical type-2 edge and we add v_{n-i} immediately before b in \prec_x . To show that Invariant 15 is maintained by this operation it is sufficient to show that b was not incident to a vertical type-2 edge before. Suppose for a contradiction that there is a vertex b' in $\{v_n, \dots, v_{n-i+1}\}$, such that (b, b') or (b', b) is a type-2 edge. As seen in the previous cases, this implies that b or b' has tag $[2, 2]$, respectively. Since in the $[2, 2]$ case the edge classified as type-2 is the one not in M and since any vertex that has tag $[2, 2]$ has label $(1, 2)$, by Lemma 13 it follows that vertical type-2 edges are chords of a cycle. Hence, b or b' would lie in the same cycle as v_{n-i} , which is a contradiction to Property 14, thus Invariant 15 holds.

Orders \prec_x and \prec_y define the placement of the vertices. By iterating over the vertices, we describe how to draw the edges to complete the drawing such that Properties (i) and (ii) are satisfied. We distinguish cases based on the tag of the current vertex v_i .

1. The tag of v_i is $[3, 1]$ or $[3, 0]$: Let $\{a, b, c\}$ be the upper critical neighbors of v_i . The construction of \prec_x ensures that not all of $\{a, b, c\}$ precede or follow v_i in \prec_x , w.l.o.g. we can assume that $a \prec_x b, v_i \prec_x c$. Then, we assign the W -port at v_i to (v_i, a) , the N -port at v_i to (v_i, b) and the E -port at v_i to (v_i, c) . If v_i has a lower critical neighbor, we assign the S -port at v_i for the edge connecting v_i to it.
2. The tag of v_i is $[2, 1]$ or $[2, 0]$: Let $\{a, b\}$ be the upper critical neighbors of v_i . We assign the N -port at v_i to (v_i, a) . Note that v_i was appended to \prec_x during its construction. If $b \prec_x v_i$, we assign the W -port at v_i to (v_i, b) . Otherwise, we assign the E -port at v_i to (v_i, b) . The S -port is assigned to the lower critical edge of v_i , if present.
3. The tag of v_i is $[1, 2]$ or $[0, 2]$: This case is symmetric to the one above by exchanging the roles of upper and lower critical neighbors and N - and S -ports.
4. The tag of v_i is $[1, 1]$: Let a be the upper critical neighbor and b the lower critical neighbor of v_i . Then we assign the N -port to the edge (v_i, a) and the S -port to (v_i, b) .
5. The tag of v_i is $[2, 2]$: Let $\{a, b\}$ and $\{c, d\}$ be the upper and lower critical neighbors of v_i . W.l.o.g. let $(v_i, a) \in M$. By Invariant 15 and construction, the edge (v_i, b) is a type-2 edge. The N - and S -ports at v_i are assigned to the edges (v_i, a) and (v_i, c) . If $d \prec_x v_i$, we assign the W -port at v_i to (v_i, d) . Otherwise, we assign the E -port at v_i to (v_i, d) .

We describe how to place the bends of the edges on each side of the box $B(u)$ of an arbitrary vertex u based on the type of the edge that is incident to u , refer to Fig. 4a. We focus on the bottom side of $B(u)$. Let (x_u, y_u) be the position of u that is defined by \prec_x and \prec_y . Recall that the box $B(u)$ has size 8×8 . Let $e = \{u, v\}$ be an edge incident to u . If e is a horizontal type-2 edge, then we place its bend at $(x_u - 3, y_u - 4)$, if $v \prec_x u$, otherwise we have $u \prec_x v$ and we place the bend at $(x_u + 3, y_u - 4)$. If e is a type-1 edge that uses the S -port of u , then segment of e incident to u passes through point $(x_u, y_u - 4)$. If e is a type-1 edge oriented from v to u such that $v \prec_y u$ and e uses either the W -port or the E -port of v , then we place the bend at $(x_u + i, y_u - 4)$ with $i \in \{-2, -1, 1, 2\}$. Since any vertex has at most four incoming type-1 edges after applying Lemma 13, we can place the bends so that no two overlap. No other edge crosses the bottom side of $B(u)$. The description for the other sides can be obtained by rotating this scheme; for the left and the right side the type-2 edges are the vertical ones.

We now describe how to draw each edge $e = (u, v)$ of G based on the relative position of u and v in \prec_x and \prec_y and the type of e . Refer to Fig. 4b. Suppose first that e is a type-2 edge. If e is a horizontal type-2 edge, then u and v are consecutive in \prec_y and $B(u)$ and $B(v)$ are aligned in y -coordinate, in particular, there is a horizontal line that contains the top side of one box and the bottom side of the other, hence it passes through the two assigned bend-points, which implies that the middle segment is horizontal. Similarly, if e is a

vertical type-2 edge, then u and v are consecutive in \prec_x by Invariant 15. Hence, the assigned points for the bends define a vertical middle segment. Suppose now that e is a type-1 edge. The case analysis for the second iteration over the vertices guarantees that for any relative position of v to u , we assigned an appropriate orthogonal port at u which allows to find a point on the first segment, such that the orthogonal middle-segment of the edge e (that is perpendicular to the first) can reach the assigned bend point on the boundary of $B(v)$.

We argue that the constructed drawing is indeed 2-bend RAC as follows. By construction, every edge consists of three segments and no bend overlaps with an edge or with another bend. Each vertical (horizontal) line either crosses only one box or contains the side of exactly two boxes, whose corresponding vertices are consecutive in \prec_x (\prec_y). This implies that if a vertical (horizontal) segment of an edge shares a point with the interior of a box, then this box correspond to one of its endvertices. Further, any oblique segment is fully contained inside the box of its endvertex, hence crossings can only happen between a vertical and a horizontal segment which implies that the drawing is RAC.

To complete the proof of Theorem 12, we discuss the time complexity and the required area. We apply Theorem 2 to $G \setminus M$ to obtain F_1, F_2, F_3 in $\mathcal{O}(n)$ time. Computing the labels clearly takes $\mathcal{O}(n)$ time. For each cycle of F_1 , the ordering of its internal vertices in Lemma 13 can be done in time linear in the size of the cycle by computing for each vertex the number of forward and backward edges, and of chords. Computing the tags takes $\mathcal{O}(n)$ time. In each of the following two iterations, we perform a constant number of operations per vertex. Hence we can conclude that the drawing can be computed in $\mathcal{O}(n)$ time. For the area, we can observe that the size of the grid defined by the boxes is $8n \times 8n$ and by construction, any vertex and any bend point is placed on a point on the grid.

► **Corollary 16.** *Given a 7-edge-colorable degree-7 graph with n vertices and a 7-edge-coloring of it, it is possible to compute in $\mathcal{O}(n)$ time a 2-bend RAC drawing of it with $\mathcal{O}(n^2)$ area.*

6 Conclusions and Open Problems

We significantly extended the previous work on RAC drawings for low-degree graphs in all reasonable settings derived by restricting the number of bends per edge to 0, 1, and 2. The following open problems are naturally raised by our work.

- Are all 4-edge-colorable degree-3 graphs RAC (refer to Question 1)?
- Are all degree-5 graphs 1-bend RAC? What about degree-6 graphs?
- Is it possible to extend Theorem 12 to all (i.e., not 7-edge-colorable) degree-7 graphs or even to (subclasses of) graphs of higher degree, e.g. Hamiltonian degree-8 graphs?
- While recognizing graphs that admit a (straight-line) RAC drawing is NP-hard [5], the complexity of the recognition problem in the 1- and 2-bend setting is still unknown.

References

- 1 Patrizio Angelini, Michael A. Bekos, Henry Förster, and Michael Kaufmann. On RAC drawings of graphs with one bend per edge. *Theor. Comput. Sci.*, 828-829:42–54, 2020. doi:10.1016/j.tcs.2020.04.018.
- 2 Patrizio Angelini, Michael A. Bekos, Julia Katheder, Michael Kaufmann, and Maximilian Pfister. RAC drawings of graphs with low degree. *CoRR*, abs/2206.14909, 2022. arXiv:2206.14909.
- 3 Patrizio Angelini, Luca Cittadini, Walter Didimo, Fabrizio Frati, Giuseppe Di Battista, Michael Kaufmann, and Antonios Symvonis. On the perspectives opened by right angle crossing drawings. *J. Graph Algorithms Appl.*, 15(1):53–78, 2011. doi:10.7155/jgaa.00217.

- 4 Evmorfia N. Argyriou, Michael A. Bekos, Michael Kaufmann, and Antonios Symvonis. Geometric RAC simultaneous drawings of graphs. *J. Graph Algorithms Appl.*, 17(1):11–34, 2013. doi:10.7155/jgaa.00282.
- 5 Evmorfia N. Argyriou, Michael A. Bekos, and Antonios Symvonis. The straight-line RAC drawing problem is NP-hard. *J. Graph Algorithms Appl.*, 16(2):569–597, 2012. doi:10.7155/jgaa.00274.
- 6 Karin Arikushi, Radoslav Fulek, Balázs Keszegh, Filip Moric, and Csaba D. Tóth. Graphs that admit right angle crossing drawings. *Comput. Geom.*, 45(4):169–177, 2012. doi:10.1016/j.comgeo.2011.11.008.
- 7 Michael A. Bekos, Walter Didimo, Giuseppe Liotta, Saeed Mehrabi, and Fabrizio Montecchiani. On RAC drawings of 1-planar graphs. *Theor. Comput. Sci.*, 689:48–57, 2017. doi:10.1016/j.tcs.2017.05.039.
- 8 Steven Chaplick, Fabian Lipp, Alexander Wolff, and Johannes Zink. Compact drawings of 1-planar graphs with right-angle crossings and few bends. *Comput. Geom.*, 84:50–68, 2019. doi:10.1016/j.comgeo.2019.07.006.
- 9 Norishige Chiba, Kazunori Onoguchi, and Takao Nishizeki. Drawing planar graphs nicely. *Acta Inform.*, 22:187–201, 1985. doi:10.1007/BF00264230.
- 10 Marek Chrobak and Thomas H. Payne. A linear-time algorithm for drawing a planar graph on a grid. *Inf. Process. Lett.*, 54(4):241–246, 1995. doi:10.1016/0020-0190(95)00020-D.
- 11 Hubert de Fraysseix, János Pach, and Richard Pollack. Small sets supporting Fáry embeddings of planar graphs. In Janos Simon, editor, *Symposium on the Theory of Computing*, pages 426–433. ACM, 1988. doi:10.1145/62212.62254.
- 12 Giuseppe Di Battista, Peter Eades, Roberto Tamassia, and Ioannis G. Tollis. *Graph Drawing: Algorithms for the Visualization of Graphs*. Prentice-Hall, 1999.
- 13 Emilio Di Giacomo, Walter Didimo, Giuseppe Liotta, and Henk Meijer. Area, curve complexity, and crossing resolution of non-planar graph drawings. *Theory Comput. Syst.*, 49(3):565–575, 2011. doi:10.1007/s00224-010-9275-6.
- 14 Walter Didimo. Right angle crossing drawings of graphs. In Seok-Hee Hong and Takeshi Tokuyama, editors, *Beyond Planar Graphs*, pages 149–169. Springer, 2020. doi:10.1007/978-981-15-6533-5_9.
- 15 Walter Didimo, Peter Eades, and Giuseppe Liotta. Drawing graphs with right angle crossings. In Frank K. H. A. Dehne, Marina L. Gavrilova, Jörg-Rüdiger Sack, and Csaba D. Tóth, editors, *Workshop on Algorithms and Data Structures*, volume 5664 of *LNCS*, pages 206–217. Springer, 2009. doi:10.1007/978-3-642-03367-4_19.
- 16 Walter Didimo, Peter Eades, and Giuseppe Liotta. A characterization of complete bipartite RAC graphs. *Inf. Process. Lett.*, 110(16):687–691, 2010. doi:10.1016/j.ipl.2010.05.023.
- 17 Walter Didimo, Peter Eades, and Giuseppe Liotta. Drawing graphs with right angle crossings. *Theor. Comput. Sci.*, 412(39):5156–5166, 2011. doi:10.1016/j.tcs.2011.05.025.
- 18 Walter Didimo and Giuseppe Liotta. The crossing-angle resolution in graph drawing. In János Pach, editor, *Thirty Essays on Geometric Graph Theory*, pages 167–184. Springer, 2013.
- 19 Walter Didimo, Giuseppe Liotta, and Fabrizio Montecchiani. A survey on graph drawing beyond planarity. *ACM Comput. Surv.*, 52(1):4:1–4:37, 2019. doi:10.1145/3301281.
- 20 Peter Eades, Antonios Symvonis, and Sue Whitesides. Three-dimensional orthogonal graph drawing algorithms. *Discret. Appl. Math.*, 103(1-3):55–87, 2000. doi:10.1016/S0166-218X(00)00172-4.
- 21 Henry Förster and Michael Kaufmann. On compact RAC drawings. In Fabrizio Grandoni, Grzegorz Herman, and Peter Sanders, editors, *European Symposium on Algorithms*, volume 173 of *LIPICs*, pages 53:1–53:21. Schloss Dagstuhl, 2020. doi:10.4230/LIPICs.ESA.2020.53.
- 22 István Fáry. On straight lines representation of planar graphs. *Acta Sci. Math. (Szeged)*, 11:229–233, 1948.
- 23 M. R. Garey and D. S. Johnson. Crossing number is NP-complete. *SIAM Journal on Algebraic Discrete Methods*, 4(3):312–316, 1983. doi:10.1137/0604033.

- 24 Ian Holyer. The NP-completeness of edge-coloring. *SIAM J. Comput.*, 10(4):718–720, 1981. doi:10.1137/0210055.
- 25 Seok-Hee Hong and Takeshi Tokuyama, editors. *Beyond Planar Graphs*. Springer, 2020. doi:10.1007/978-981-15-6533-5.
- 26 Weidong Huang, Peter Eades, and Seok-Hee Hong. Larger crossing angles make graphs easier to read. *J. Vis. Lang. Comput.*, 25(4):452–465, 2014. doi:10.1016/j.jvlc.2014.03.001.
- 27 Julius Petersen. Die Theorie der regulären graphs. *Acta Mathematica*, 15:193–220, 1891. doi:10.1007/BF02392606.
- 28 Helen C. Purchase. Effective information visualisation: a study of graph drawing aesthetics and algorithms. *Interact. Comput.*, 13(2):147–162, 2000. doi:10.1016/S0953-5438(00)00032-1.
- 29 Marcus Schaefer. Rac-drawability is \exists R-Complete. In Helen C. Purchase and Ignaz Rutter, editors, *Graph Drawing and Network Visualization*, volume 12868 of *LNCIS*, pages 72–86. Springer, 2021. doi:10.1007/978-3-030-92931-2_5.
- 30 Sherman K. Stein. Convex maps. *Proc. American Math. Soc.*, 2(3):464–466, 1951.
- 31 E. Steinitz and H. Rademacher. *Vorlesungen über die Theorie der Polyeder*. Julius Springer, Berlin, Germany, 1934.
- 32 Roberto Tamassia. On embedding a graph in the grid with the minimum number of bends. *SIAM J. Comput.*, 16(3):421–444, 1987. doi:10.1137/0216030.
- 33 William Thomas Tutte. How to draw a graph. *Proc. London Math. Soc.*, 13:743–768, 1963.
- 34 Klaus Wagner. Bemerkungen zum Vierfarbenproblem. *Jahresbericht der Deutschen Mathematiker-Vereinigung*, 46:26–32, 1936.