Journal of Graph Algorithms and Applications
http://jgaa.info/ vol. 0, no. 0, pp. 0–0 (0)
DOI:

Simultaneous Drawing of Planar Graphs with Right-Angle Crossings and Few Bends

Michael A. Bekos [1] *Thomas C. van Dijk* [2] *Philipp Kindermann* [2]
Alexander Wolff [2]

[1]Wilhelm-Schickard-Institut für Informatik, Universität Tübingen, Germany.
bekos@informatik.uni-tuebingen.de
[2]Lehrstuhl für Informatik I, Universität Würzburg, Germany.
http://www1.informatik.uni-wuerzburg.de/en/staff

Abstract

Given two planar graphs that are defined on the same set of vertices, a *RAC simultaneous drawing* is a drawing of the two graphs where each graph is drawn planar, no two edges overlap, and edges of one graph can cross edges of the other graph only at right angles. In the geometric version of the problem, vertices are drawn as points and edges as straight-line segments. It is known, however, that even pairs of very simple classes of planar graphs (such as wheels and matchings) do not always admit a geometric RAC simultaneous drawing.

In order to enlarge the class of graphs that admit RAC simultaneous drawings, we allow edges to have bends. We prove that any pair of planar graphs admits a RAC simultaneous drawing with at most six bends per edge. For more restricted classes of planar graphs (e.g., matchings, paths, cycles, outerplanar graphs, and subhamiltonian graphs), we significantly reduce the required number of bends per edge. All our drawings use quadratic area.

Submitted:	Reviewed:	Revised:	Accepted:	Final:
		Published:		
	Article type:		Communicated by:	

A shorter version of this work appeared in the Proceedings of the 9th International Workshop on Algorithms and Computation (WALCOM'15) [6]. This research was supported by the ESF EuroGIGA project GraDR (DFG grant Wo 758/5-1). The work of M.A. Bekos is implemented within the framework of the Action "Supporting Postdoctoral Researchers" of the Operational Program "Education and Lifelong Learning" (Action's Beneficiary: General Secretariat for Research and Technology), and is co-financed by the European Social Fund (ESF) and the Greek State.

E-mail addresses: bekos@informatik.uni-tuebingen.de (Michael A. Bekos) thomas.van.dijk@uni-wuerzburg.de (Thomas C. van Dijk) philipp.kindermann@uni-wuerzburg.de (Philipp Kindermann) alexander.wolff@uni-wuerzburg.de (Alexander Wolff)

1 Introduction

A simultaneous embedding of two planar graphs embeds each graph in a planar way—using the same vertex positions for both embeddings. Edges of one graph are allowed to intersect edges of the other graph. There are two versions of the problem: In the first version, called *Simultaneous Embedding with Fixed Edges* (SEFE), edges that occur in both graphs must be embedded in the same way in both graphs (and hence, cannot be crossed by any other edge). In the second version, called *Simultaneous Embedding*, these edges can be drawn differently for each of the graphs. Both versions of the problem have a geometric variant where edges must be drawn using straight-line segments.

Simultaneous embedding problems have been extensively investigated over the last few years, starting with the work of Brass et al. [10] on simultaneous straight-line drawing problems. Bläsius et al. [8] recently surveyed the area. For example, it is possible to decide in linear time whether a pair of graphs admits a SEFE or not, if the common graph is biconnected [2] or has a fixed planar embedding [1]. Further, SEFE can be decided in polynomial time if each connected component of the common graph is biconnected or subcubic [28], or outerplanar with cutvertices of degree at most 3 [9].

When actually drawing these simultaneous embeddings, a natural choice is to use straight-line segments. In general, it is NP-hard to decide whether two planar graphs admit a geometric simultaneous embedding [16]. This negative results also holds even if one of the input graphs is a matching [12]. Only very few graphs can be drawn in this way, and some existing results require exponential area. For instance, there exist a tree and a path which cannot be drawn simultaneously with straight-line segments [3], and the algorithm for simultaneously drawing a tree and a matching [12] does not provide a polynomial area bound.

For the case of edges with bends (that is, polygonal edges), Haeupler et al. [19] showed that a drawing with no bends in one of the input graphs and at most as many bends per edge as the number of common vertices in the other input graph can be found if the common graph is biconnected. Erten and Kobourov [15] showed that three bends per edge and quadratic area suffice for any pair of planar graphs (without fixed edges), and that one bend per edge suffices for pairs of trees. Kammer [24] reduced the required number of bends per edge to two for the general case of planar graphs. Grilli et al. [18] proved that every SEFE embedding of two graphs admits a drawing with no bends per common edge and at most nine bends per exclusive edge. If the common graph is biconnected, they can reduce the number of bends per exclusive edge to three. In the aforementioned results, however, the *crossing angles* can be very small.

In this paper, we suggest a new approach that overcomes the aforementioned problems. We insist that crossings occur at right angles, thereby "taming" them. We do this while drawing all vertices and all bends on an integer grid of size $O(n) \times O(n)$ for any n-vertex graph, and we can still draw any pair of planar graphs simultaneously. We do not consider the problem of fixed edges. In a way, our results give a measure for the geometric complexity of simultaneous embeddability for various pairs of graph classes, some of which can be combined more easily (that is, with fewer bends) and some not as easily (that is, with more bends).

More formally, let $G_1 = (V, E_1)$ and $G_2 = (V, E_2)$ be two planar graphs defined

Table 1. A short summary of our results

Graph classes			Number of bends	Ref.
planar	$+$	planar	$6+6$	Thm. 1
subhamiltonian	$+$	subhamiltonian	$4+4$	Cor. 1
outerplanar	$+$	outerplanar	$3+3$	Thm. 2
cycle	$+$	cycle	$1+1$	Thm. 3
caterpillar	$+$	cycle	$1+1$	Thm. 4
four matchings			$1+1+1+1$	Thm. 5
tree	$+$	matching	$1+0$	Thm. 6
wheel	$+$	matching	$2+0$	Thm. 7
outerpath	$+$	matching	$2+1$	Thm. 8

on the same vertex set. We say that G_1 and G_2 admit a *RAC simultaneous drawing* (or, RACSIM *drawing* for short), if we can place the vertices on the plane such that:

(i) each edge is drawn as a polyline,

(ii) each graph is drawn planar,

(iii) there are no edge overlaps, and

(iv) crossings between edges in E_1 and E_2 occur at right angles.

Argyriou et al. [4] introduced and studied the geometric version of the RACSIM drawing problem. In particular, they proved that any pair of a cycle and a matching admits a geometric RACSIM drawing on an integer grid of quadratic size, while there exists a pair of a wheel and a cycle that does not admit a geometric RACSIM drawing. The problem that we study was left as an open problem.

Closely related to the RACSIM drawing problem is the problem of simultaneously drawing a (primal) embedded graph and its dual, so that the primal-dual edge crossings form right angles. Brightwell and Scheinermann [11] proved that this problem always admits a solution, if the input graph is triconnected planar. Erten and Kobourov [14] presented an $O(n)$-time algorithm that computes simultaneous drawings of a triconnected planar graph and its dual on an integer grid of size $O(n) \times O(n)$, where n is the total number of vertices in the graph and its dual; their drawings, however, can have non-right angle crossings.

Our contribution: Our main result is that any pair of planar graphs admits a RACSIM drawing with at most six bends per edge. For pairs of subhamiltonian graphs and pairs of outerplanar graphs, we can reduce the required number of bends per edge to four and three, respectively; see Section 2. (Recall that a subhamiltonian graph is a subgraph of a planar Hamiltonian graph.) Then, we turn our attention to pairs of more restricted graph classes where we can guarantee RACSIM drawings with one bend per edge or two bends per edge; see Sections 3 and 4, respectively. Table 1 summarizes our results. Note that all our algorithms run in linear time, with the exception of the algorithm for an outerpath and a matching (see Theorem 8) that runs in $O(n \log n)$ time. The produced drawings fit on integer grids of quadratic size. The main approach of all our algorithms is to find linear orders on the vertices of the two graphs and then to compute the exact coordinates of the vertices of both graphs based on these orders. Although the produce drawings may contain non-rectilinear edge-segments (referred to as *slanted* segments, for short), all crossings in our drawings appear between horizontal and vertical edge segments.

2 RacSim Drawings of general graphs

In this section, we study general planar graphs and show how to efficiently construct RacSim drawings with few bends per edge in quadratic area. We prove that two planar graphs on a common set of vertices admit a RacSim drawing with six bends per edge (Theorem 1). For pairs of subhamiltonian graphs, we lower the required number of bends per edge to 4 (Corollary 1) and for pairs of outerplanar graphs to 3 (Theorem 2). Note that the class of subhamiltonian graphs is equivalent to the class of 2-page book-embeddable graphs, and the class of outerplanar graphs is equivalent to the class of 1-page book-embeddable graphs [7].

Central to our approach is an algorithm by Kaufmann and Wiese [25] that embeds any planar graph such that vertices are mapped to points on a horizontal line (so-called *spine*) and each edge crosses the spine at most once; see Figure 1a. Clearly, if one replaces each spine crossing with a dummy vertex, then a *linear order* of the (original and dummy) vertices is obtained with the property that every edge is either completely above or completely below the spine. In order to determine the exact locations of the vertices of the two given graphs in our problem, we basically employ the linear order induced by the first graph to compute the x-coordinates and the linear order induced by the second graph to compute the corresponding y-coordinates. (Note that the aforementioned approach has been used for simultaneous drawing problems before [15].) Then, we draw the edges of both graphs, so that all edges-crossing(i) appear in the interior of the smallest axis-aligned rectangle containing all vertices, and (ii) are restricted between vertical and horizontal edge-segments, that is, slanted edge-segments are crossing-free.

Theorem 1 *Two planar graphs on a common set of n vertices admit a* RacSim *drawing on an integer grid of size $(14n - 26) \times (14n - 26)$ with six bends per edge. The drawing can be computed in $O(n)$ time.*

Proof: Let $\mathcal{G}_1 = (V, E_1)$ and $\mathcal{G}_2 = (V, E_2)$ be the given planar graphs. For $m = 1, 2$, let ξ_m be an embedding of $\mathcal{G}_m$ according to the algorithm of Kaufmann and Wiese [25]. As already stated, we subdivide all edges of $\mathcal{G}_m$ that cross the spine in ξ_m, by introducing a single dummy vertex for each such edge at the point where it crosses the spine. Let $\mathcal{G}'_m = (V'_m, E'_m) = (V \cup V_m, A_m \cup B_m)$ be the resulting graph, where V_m is the set of dummy vertices of $\mathcal{G}'_m$, A_m is the set of edges that are drawn completely above the spine and B_m is the set of edges that are drawn completely below the spine. Let also ξ'_m be the embedding of $\mathcal{G}'_m$.

In the following, we show how to determine the x-coordinates of the vertices in V_1'; the y-coordinates of the vertices in V_2' are determined analogously. Let n_1' be the number of vertices in V_1' and let $v_1 \rightarrow v_2 \rightarrow \cdots \rightarrow v_{n_1'}$ be the linear order of the vertices of V_1' along the spine in ξ_1'. We place v_1 in the leftmost available column. Between any two consecutive vertices v_i and v_{i+1}, we reserve several columns for the bends of the edges incident to v_i and v_{i+1}, in the following order; see Figure 1b:

(i) a column for the first bend on all edges leaving v_i in A_2;

(ii) a column for each edge $(v_i, v_j) \in E_1'$ with $j > i$;

(iii) a column for each edge $(v_k, v_{i+1}) \in E_1'$ with $k \leq i$;

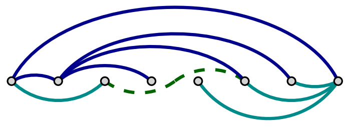
(iv) a column for the last bend on all edges entering v_{i+1} in B_2.

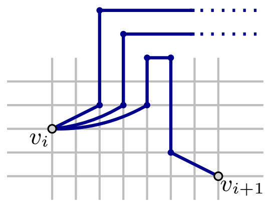
Note that we can save some columns among those reserved for (ii) and (iii), because an edge in A_1 and an edge in B_1 can use the same column for their bend. With the aforementioned procedure we fully specify the x-coordinates of the vertices in V_1'; the y-coordinates of the vertices in V_2' are determined analogously. Let, now, R be the smallest axis-aligned rectangle that contains all vertices of the common vertex set of $\mathcal{G}_1'$ and $\mathcal{G}_2'$ (refer to the gray-colored rectangle of Figure 1c). Then, the y-coordinates of the dummy vertices of V_1' and the x-coordinates of the dummy vertices of V_2' —that so far have not been determined by our algorithm—can be arbitrarily selected, as long as the corresponding vertices are inside R.

We proceed to describe how to draw the edges of graph $\mathcal{G}_1'$ with at most four bends per edge such that all edge segments of $\mathcal{G}_1'$ in R are either vertical or of y-length exactly 1; see Figure 1d. The edges of graph $\mathcal{G}_2$ are drawn analogously. First, we draw the edges $(v_i, v_j) \in A_1$ with $i < j$ in a nested order: When we draw the edge (v_i, v_j), there is no edge $(v_k, v_l) \in A_1$ with $k \leq i$ and $l \geq j$ that has not already been drawn. Recall that the first column to the right and the first column to the left of every vertex is reserved for the edges in E_1; hence, we assume that they are already used. We draw (v_i, v_j) with at most four bends as follows. We start with a slanted segment incident to v_i that has its other endpoint in the row above v_i and in the first unused column that does not lie to the left of v_i. We follow with a vertical segment to the top that leaves R. We add a horizontal segment above R. In the last unused column that does not lie to the right of v_j, we add a vertical segment that ends one row above v_j. We close the edge with a slanted segment that has its endpoint in v_j. We draw the edges in B_1 symmetrically with the horizontal segment below R.

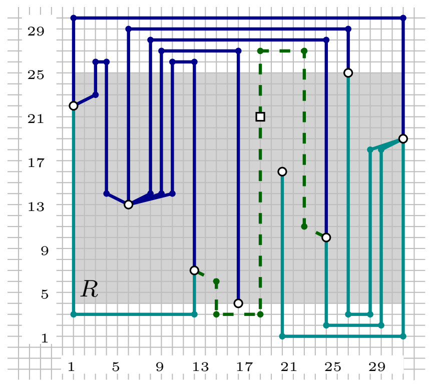
Note that this algorithm always uses the top and the bottom port of a vertex v, if there is at least one edge incident to v in A_1 and B_1, respectively. Since there is exactly one edge incident to each dummy vertex in A_1 and B_1, respectively, the edges incident to a dummy vertex only use the top and the bottom port. We create a drawing of $\mathcal{G}_1$ and $\mathcal{G}_2$ with at most 6 bends per edge by removing (or, more precisely, by smoothing out) the dummy vertices from the drawing.

By construction, all edge segments of E_1 inside R are either vertical segments or slanted segments of y-length 1. Symmetrically, all segments of E_2 inside R are either horizontal segments or slanted segments of x-length 1. Thus, the slanted segments cannot intersect. Furthermore, all crossings inside R occur between a horizontal and a

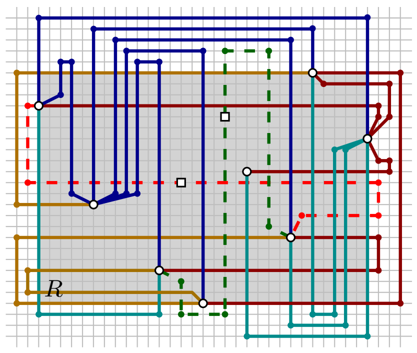

(a) A drawing of a planar graph by Kaufmann and Wiese [25]



(b) Reserving additional columns between two vertices



(c) The graph in (a) drawn by our algorithm with at most six bends per edge.



(d) The RACSIM drawing of two planar graphs by our algorithm

Figure 1. A RACSIM drawing of two planar graphs. The edges that cross the spine are drawn dashed; the dummy vertices on these edges are drawn as squares.

vertical segment, and thus form right angles. Also, there are no segments in E_1 that lie to the left or to the right of R, and there are no segments in E_2 that lie above or below R. Hence, there are no crossings outside R, which guarantees that the constructed drawing of $\mathcal{G}_1$ and $\mathcal{G}_2$ is a RACSIM drawing.

We now count the columns used by the drawing. For the leftmost and the rightmost vertex, we reserve one additional column for its incident edges in E_2; for the remaining vertices, we reserve two such columns. For each edge in E_1, we need up to three columns: one for each endpoint of the slanted segment at each vertex and one for the vertical segment that crosses the spine, if it exists. Note that at least one edge per vertex does not need a slanted segment. For each edge in E_2, we need at most one column for the vertical segment to the side of R. Since there are at most $3n-6$ edges, we need at most $3n-2+3\cdot(3n-6)-n+3n-6=14n-26$ columns. By symmetry, we need the same number of rows.

The algorithm of Kaufmann and Wiese that computes a drawing with at most 3 bends per edge runs in $O(n)$ time. We can compute the nested order of the edges in linear time from the embedding, as the circular order of the edges around a vertex gives a hierarchical order on the edges that describes the nested order of the edges. Thus, our algorithm also runs in $O(n)$ total time. $\square$

We can improve the result of Theorem 1 for subhamiltonian graphs, which admit 2-page book embeddings, in which no edges cross the spine [7]. Since such edges are the only ones that need six bends, we can reduce the number of bends per edge to four. Further, the number of columns and rows are reduced by one per edge. This is summarized in the following corollary. Note, however, that it is NP-hard to decide whether a given planar graph is subhamiltonian, even for maximal planar graphs [29]. A 2-page book embedding can be found in linear time, if the ordering of the vertices along the spine is given [20]. However, several graphs are (sub)hamiltonian, for example 4-connected planar graphs [27], planar graphs without separating triangles [23], Halin graphs [13], planar graphs with maximum degree 3 or 4 [22, 5].

Corollary 1 *Two subhamiltonian graphs on a common set of n vertices admit a* RACSIM *drawing on an integer grid of size $(11n - 32) \times (11n - 32)$ with four bends per edge. The algorithm runs in $O(n)$ time if the subhamiltonian cycles of both graphs are given.*

In order to get a RACSIM drawing of two outerplanar graphs, we use a decomposition of each of the graphs into two forests. The following lemma shows that we can do this in linear time.

Lemma 1 *Every outerplanar graph can be decomposed into two forests. This decomposition can be computed in linear time.*

Proof: It follows by Nash-Williams' formula [26] that every outerplanar graph has arboricity 2, that is, it can be decomposed into two forests. To prove the linear running time, we first assume biconnectivity and augment the input graph to a maximal outerplanar graph. Now, if we add a new vertex that is incident to all vertices of the graph, the result is a maximal planar graph which can be decomposed into three trees [17], so that one of them is a star incident to the newly added vertex. Hence, the removal of this vertex yields a decomposition into two trees. The desired decomposition into two forests follows from the removal of the edges added to augment the graph to maximal outerplanar. For an input outerplanar graph that is not biconnected, we have to compute the aforementioned decomposition for each of its biconnected components, which form a tree (so-called *BC-tree*) and therefore do not affect the structure of the overall decomposition, that is, it still consists of two forests. Since both the decomposition of a maximal planar graph into three trees and the computation of the biconnected components of the input outerplanar graph can be done in linear time, it follows that overall the decomposition can done in linear time. □

With this lemma, we can further improve the required number of bends per edge to three for outerplanar graphs. (Recall that all outerplanar graphs are 1-page book embeddable.) We will use the order of the vertices on the spine of a 1-page book embedding to compute a 2-page book embedding, in which every edge uses a rectilinear port at one of its endpoints and thus omits one of its bends.

Theorem 2 *Two outerplanar graphs on a common set of n vertices admit a* RACSIM *drawing on an integer grid of size $(7n - 10) \times (7n - 10)$ with three bends per edge. The drawing can be computed in $O(n)$ time.*

Proof: Let $\mathcal{O}_1 = (V, E_1)$ and $\mathcal{O}_2 = (V, E_2)$ be the given outerplanar graphs. We will embed $\mathcal{O}_1$ and $\mathcal{O}_2$ on two pages with one forest per page.

To do so, we first create 1-page book embeddings for $\mathcal{O}_1$ and $\mathcal{O}_2$ using the linear time algorithm of Heath [21]. This implies the orders of the vertices of both graphs along the spine. It follows by Corollary 1 that, by using the algorithm described in the proof of Theorem 1, we can create a RACSIM drawing of $\mathcal{O}_1$ and $\mathcal{O}_2$ with at most four bends per edge. We will now show how to adjust the algorithm to reduce the number of bends by one.

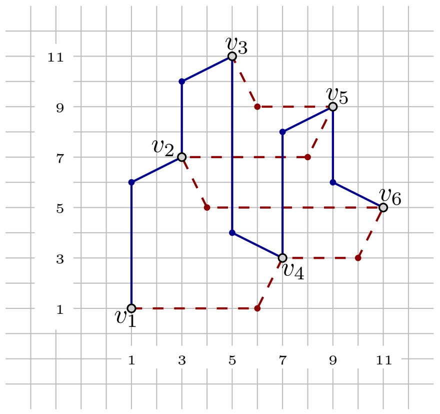
Let A_1 and B_1 be the two forests $\mathcal{O}_1$ is decomposed into according to Lemma 1. We will draw the edges of A_1 above the spine and the edges B_1 below the spine. By rooting the trees in A_1 in arbitrary vertices, we can direct each edge such that every vertex has exactly one incoming edge. Recall that, in the drawing produced in Theorem 1, one edge per vertex can use its top port. We adjust the algorithm such that every directed edge (v, w) enters vertex w from its top port. To do so, we draw the edge as follows. We start with a slanted segment of y-length exactly 1. We follow with a vertical segment to the top. We proceed with a horizontal segment that ends directly above w and finish the edge with a vertical segment that enters w from the top port. We use the same approach for the edges in B_1, using the bottom port. We treat the second outerplanar graph $\mathcal{O}_2$ analogously.

Since every port of a vertex is only used once, the drawing has no overlaps. We now analyze the number of columns used. For every vertex except for the leftmost and rightmost, we again reserve two additional columns for the edges in E_2; for the remaining two vertices, we reserve one additional column. However, the edges in E_1 now only need one column for the bend of the single slanted segment. For every edge in E_2, we need up to one column for the vertical segment to the side of R. Since there are at most $2n - 4$ edges, our drawing needs $3n - 2 + 2n - 4 + 2n - 4 = 7n - 10$ columns. Analogously, we can show that the algorithm needs $7n - 10$ rows. Since the decomposition can be computed in $O(n)$ time, our algorithm also requires $O(n)$ time. $\square$

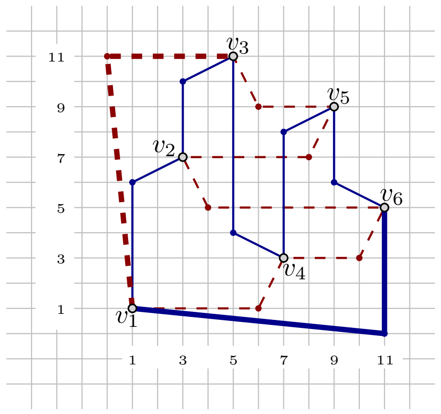
3 RACSIM Drawings with one bend per edge

In this section, we study simple classes of planar graphs and show how to efficiently construct RACSIM drawings with one bend per edge in quadratic area. In particular, we prove that two cycles or four matchings on a common set of n vertices admit a RACSIM drawing on an integer grid of size $2n \times 2n$; see Theorems 3 and 5, respectively. If the input to our problem is a caterpillar and a cycle, then we can compute a RACSIM drawing with one bend per edge on an integer grid of size $(2n-1) \times 2n$; see Theorem 4. For a tree and a cycle, we can construct a RACSIM drawing with one bend per tree edge and no bends in the matching edges on an integer grid of size $n \times (n - 1)$; see Theorem 6.

Lemma 2 *Two paths on a common set of n vertices admit a* RACSIM *drawing on an integer grid of size $2n \times 2n$ with one bend per edge. The drawing can be computed in $O(n)$ time.*



(a) Two paths: $\mathcal{P}_1$ (solid) and $\mathcal{P}_2$ (dashed)



(b) Two cycles: $\mathcal{C}_1$ (solid) and $\mathcal{C}_2$(dashed). The closing edges are drawn bold.

Figure 2. RACSIM drawings with one bend per edge

Proof: Let $\mathcal{P}_1 = (V, E_1)$ and $\mathcal{P}_2 = (V, E_2)$ be the two input paths. Following standard practices from the literature (see, for example, Brass et al. [10]), we draw $\mathcal{P}_1$ x-monotone and $\mathcal{P}_2$ y-monotone. This ensures that the drawing of both paths will be planar. We will now describe how to compute the exact coordinates of the vertices and how to draw the edges of $\mathcal{P}_1$ and $\mathcal{P}_2$, such that all crossings are at right angles and, more importantly, no edge segments overlap.

For $m = 1, 2$ and any vertex $v \in V$, let $\pi_m(v)$ be the position of v in $\mathcal{P}_m$. Then, v is drawn at the point $p(v) = (2\pi_1(v) - 1, 2\pi_2(v) - 1)$; see Figure 2a. It remains to determine, for each edge $e = (v, v')$, where it bends. First, assume that $e \in E_1$ and e is directed from its left endpoint, say v, to its right endpoint, say v'. Then, we place the bend at $p(v') - (2, \text{sgn}(y(v') - y(v)))$. Second, assume that $e \in E_2$ and e is directed from its bottom endpoint, say v, to its top endpoint, say v'. Then, we place the bend at $p(v') - (\text{sgn}(x(v') - x(v)), 2)$.

Clearly, the area required by the drawing is $(2n - 1) \times (2n - 1)$. An edge of $\mathcal{P}_1$ leaves its left endpoint vertically and enter its right endpoint with a slanted segment of x-length 1 and y-length 2. Similarly, an edge of $\mathcal{P}_2$ leaves its bottom endpoint horizontally and enters its top endpoint with a slanted segment of x-length 2 and y-length 1. Hence, the slanted segments cannot be involved in crossings or overlaps. Since $\mathcal{P}_1$ and $\mathcal{P}_2$ are x- and y-monotone, respectively, it follows that all crossings must involve a vertical edge segment of $\mathcal{P}_1$ and a horizontal edge segment of $\mathcal{P}_2$, which clearly yields right angles at the crossing points. $\square$

We say that an edge uses the bottom/left/right/top *port* of a vertex if it enters the vertex from the bottom/left/right/top.

Theorem 3 *Two cycles on a common set of n vertices admit a* RACSIM *drawing on an integer grid of size $2n \times 2n$ with at most one bend per edge. The drawing can be computed in $O(n)$ time.*

Proof: Let $\mathcal{C}_1 = (V, E_1)$ and $\mathcal{C}_2 = (V, E_2)$ be the two input cycles, and let $v \in V$ be an arbitrary vertex. We temporarily delete one edge $(v, w_1) \in E_1$ from $\mathcal{C}_1$ and one edge $(v, w_2) \in E_2$ from $\mathcal{C}_2$ (refer to the bold-drawn edges of Figure 2b). This results into two paths $\mathcal{P}_1 = \langle v, \dots, w_1 \rangle$ and $\mathcal{P}_2 = \langle v, \dots, w_2 \rangle$. We employ the algorithm desribed in Lemma 2 to construct a RACSIM drawing of $\mathcal{P}_1$ and $\mathcal{P}_2$ on an integer grid of size $(2n-1) \times (2n-1)$. Since v is the first vertex in both paths, it is placed at the bottom-left corner of the bounding box containing the drawing. Since w_1 and w_2 are the last vertices in $\mathcal{P}_1$ and $\mathcal{P}_2$, respectively, w_1 is placed on the right side, and w_2 on the top side of the bounding box. By construction, the bottom port of w_1 and the left port of w_2 are both unoccupied. Hence, the edges (v, w_1) and (v, w_2) that form $\mathcal{C}_1$ and $\mathcal{C}_2$ can be drawn with a single bend at points $(2n-1, 0)$ and $(0, 2n-1)$, respectively; see Figure 2b. Since both edges are in the exterior of the bounding box containing the drawing, none of them is involved in crossings. On the other hand, the total area of the drawing gets larger by a single unit in each dimension. □

Theorem 4 *A caterpillar and a cycle on a common set of n vertices admit a* RACSIM *drawing on an integer grid of size $(2n-1) \times 2n$ with one bend per edge. The drawing can be computed in $O(n)$ time.*

Proof: We denote by $\mathcal{A} = (V, E_\mathcal{A})$ the caterpillar and by $\mathcal{C} = (V, E_\mathcal{C})$ the cycle. A caterpillar can be decomposed into a path, called *spine*, and a set of leaves connected to the path, called *legs*. Let $v_1, v_2, \dots, v_n$ be the vertex set of $\mathcal{A}$ ordered as follows; see Figure 3: Starting from an endpoint of the spine of $\mathcal{A}$, we traverse the spine such that we first visit all legs incident to a spine vertex before moving on to the next spine vertex. This order defines the x-order of the vertices in the output drawing.

As in the proof of Theorem 3, we temporarily delete an edge of $\mathcal{C}$ incident to v_1 (see the bold dashed edge in Figure 3) and obtain a path which we denote by $\mathcal{P} = (V, E_\mathcal{P})$. For any vertex $v \in V$, let $\pi(v)$ be the position of v in $\mathcal{P}$. The map π determines the y-order of the vertices in our drawing. For $i = 1, 2, \dots, n$, we draw vertex v_i at point $p(v_i) = (2i-1, 2\pi(v_i)-1)$. It remains to determine, for each edge $e = (v, v')$, where it bends. First, assume that $e \in E_\mathcal{P}$ and e is directed from its bottom endpoint, say v, to its top endpoint, say v' (see the thin dashed edges in Figure 3). Then, we place the bend at $p(v) + (\operatorname{sgn}(x(v') - x(v)), 2)$. Second, assume that $e \in E_\mathcal{A}$ and e is directed from its left endpoint, say v, to its right endpoint, say v' (see the solid edges in Figure 3). Then, we place the bend at $(x(v'), y(v) + \operatorname{sgn}(y(v') - y(v)))$.

The approach described above ensures that $\mathcal{P}$ is drawn y-monotone, hence planar. The spine of $\mathcal{A}$ is drawn x-monotone. The legs of a spine vertex of $\mathcal{A}$ are drawn to the right of their parent spine vertex and to the left of the next vertex along the spine. Hence, $\mathcal{A}$ is drawn planar as well. The slanted segments of $\mathcal{A}$ are of y-length 1, while the slanted segments of $\mathcal{P}$ are of x-length 1. Thus, they cannot be involved in crossings, which implies that all crossings form right angles.

It remains to draw the edge e in $E_\mathcal{C} \setminus E_\mathcal{P}$. Recall that e is incident to v_1, which lies at the bottom-left corner of the bounding box containing our drawing. Let v_j be the other endpoint of e. Since $\pi(v_j) = n$, vertex v_j lies at the top side of the bounding box. As the top port of v_1 is not used, we can draw the first segment of e vertical, bending at $(1, 2n)$; see the bold dashed edge in Figure 3.

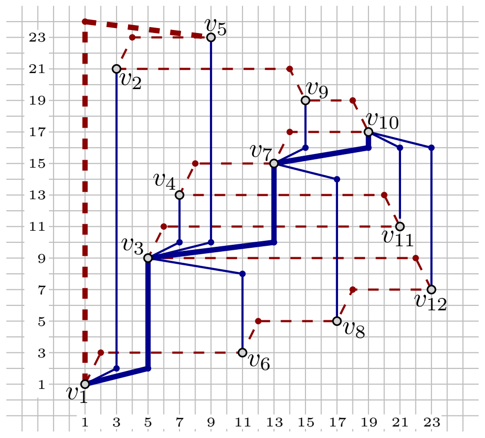


Figure 3. A RacSim drawing of a caterpillar (solid; its spine is drawn bold) and a cycle (dashed)

Clearly, the total area required by the drawing is $(2n-1) \times 2n$. □

Theorem 5 *Four matchings on a common set of n vertices admit a* RacSim *drawing on an integer grid of size $2n \times 2n$ with at most one bend per edge. The drawing can be computed in $O(n)$ time.*

Proof: Let $\mathcal{M}_1 = (V, E_1)$, $\mathcal{M}_2 = (V, E_2)$, $\mathcal{M}_3 = (V, E_3)$ and $\mathcal{M}_4 = (V, E_4)$ be the input matchings. Without loss of generality, we assume that all matchings are perfect; otherwise, we augment them to perfect matchings. Let $\mathcal{M}_{1,2} = (V, E_1 \cup E_2)$ and $\mathcal{M}_{3,4} = (V, E_3 \cup E_4)$. Since $\mathcal{M}_1$ and $\mathcal{M}_2$ are defined on the same vertex set, $\mathcal{M}_{1,2}$ is a 2-regular graph. Thus, each connected component of $\mathcal{M}_{1,2}$ corresponds to a cycle of even length which alternates between edges of $\mathcal{M}_1$ and $\mathcal{M}_2$; see Figure 4. The same holds for $\mathcal{M}_{3,4}$. We will determine the x-coordinates of the vertices from $\mathcal{M}_{1,2}$, and the y-coordinates from $\mathcal{M}_{3,4}$.

We start with choosing an arbitrary vertex $v \in V$. Let $\mathcal{C}$ be the cycle of $\mathcal{M}_{1,2}$ containing vertex v. We determine the x-coordinates of the vertices of $\mathcal{C}$ by traversing $\mathcal{C}$ in some direction, starting from vertex v. For each vertex u in $\mathcal{C}$, let $\pi_1(u)$ be the discovery time of u according to this traversal, with $\pi_1(v) = 0$. Then, we set $x(u) = 2\pi_1(u) + 1$. Next, we determine the y-coordinates of the vertices of all cycles $\mathcal{C}_1, \ldots, \mathcal{C}_k$ of $\mathcal{M}_{3,4}$ that have at least one vertex with a determined x-coordinate, ordered as follows. For $i = 1, \ldots, k$, let a_i be the *anchor* of $\mathcal{C}_i$, that is, the vertex with the smallest determined x-coordinate of all vertices in $\mathcal{C}_i$. Then, $x(a_1) < \ldots < x(a_k)$. In what follows, we start with the first cycle $\mathcal{C}_1$ of the computed order and determine the y-coordinates of its vertices. To do so, we traverse $\mathcal{C}_1$ in some direction, starting from its anchor vertex a_1. For each vertex u in $\mathcal{C}_1$, let $\pi_2(u)$ be the discovery time of u according to this traversal, with $\pi_2(a_1) = 0$. Then, we set $y(u) = 2\pi_2(u) + 1$. We proceed analogously with the remaining cycles $\mathcal{C}_i$, $i = 2, \ldots, k$, setting $\pi_2(a_i) = \max_{u \in \mathcal{C}_{i-1}} \pi_2(u) + 1$.

Now, there are no vertices with only one determined x-coordinate. However, there might exist vertices with only one determined y-coordinate. If this is the case, we re-

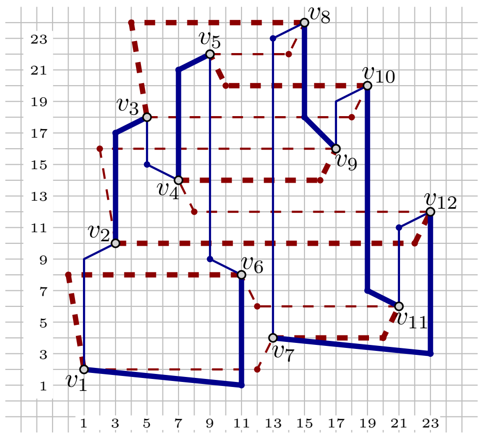


Figure 4. A RacSim drawing of four matchings $\mathcal{M}_1$ (solid-plain), $\mathcal{M}_2$ (solid-bold), $\mathcal{M}_3$ (dashed-plain) and $\mathcal{M}_4$ (dashed-bold)

peat the aforementioned procedure to determine the x-coordinates of the vertices of all cycles of $\mathcal{M}_{1,2} \setminus \mathcal{C}$ that have at least one vertex with a determined y-coordinate, but without determined x-coordinates. If there are no vertices with only one determined coordinate left, either all coordinates are determined, or we restart this procedure with another arbitrary vertex that has no determined coordinates. Thus, our algorithm guarantees that the x- and y-coordinate of all vertices are eventually determined.

Note that, for each cycle in $\mathcal{M}_{1,2}$, there is exactly one edge $e = (v, v')$, called *closing edge*, with $\pi_1(v') > \pi_1(v) + 1$. Analogously, for each cycle in $\mathcal{M}_{3,4}$, there is exactly one closing edge $e = (u, u')$ with $\pi_2(u') > \pi_2(u) + 1$.

It remains to determine, for each edge $e = (v, v')$, where it bends. First, assume that $e \in E_1 \cup E_2$ and e is directed from its left endpoint, say v, to its right endpoint, say v'. If e is not a closing edge, we place the bend at $(x(v') - 2, y(v') - \operatorname{sgn}(y(v') - y(v))$. Otherwise, we place the bend at $(x(v'), y(v) - 1)$. Second, assume that $e \in E_3 \cup E_4$ and e is directed from its bottom endpoint, say v, to its top endpoint, say v'. If e is not a closing edge, we place the bend at $(x(v') - \operatorname{sgn}(x(v') - x(v)), x(v') - 2)$. Otherwise, we place the bend at $(x(v) - 1, y(v'))$; see Figure 4.

Our choice of coordinates guarantees that the x-coordinates of the cycles of $\mathcal{M}_{1,2}$ and the y-coordinates of the cycles of $\mathcal{M}_{3,4}$ form disjoint intervals. Thus, the area below a cycle of $\mathcal{M}_{1,2}$ and the area to the left of a cycle of $\mathcal{M}_{3,4}$ are free from vertices. Hence, the slanted segments of the closing edges cannot have a crossing that violates the RAC restriction. The total area required by the drawings is $2n \times 2n$. □

Theorem 6 *A tree and a matching on a common set of n vertices admit a* RacSim *drawing on an integer grid of size $n \times (n-1)$ with one bend per tree-edge, and no bends in the edges of the matching. The drawing can be computed in $O(n)$ time.*

Proof: We use an algorithm inspired by the algorithm for drawing a geometric simultaneous embedding of a tree and a matching by Cabello et al. [12]; see Figure 5 for an example drawing. First, we root the tree in an arbitrary leaf, getting a directed tree $\mathcal{T}$.

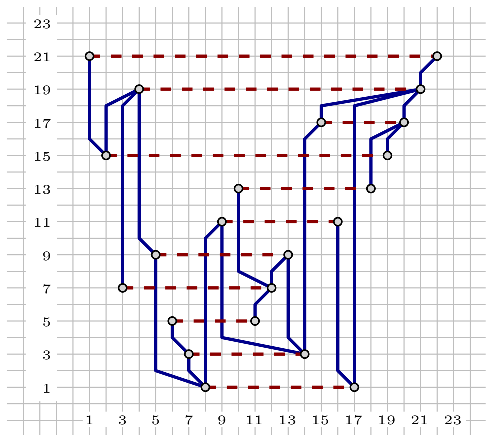


Figure 5. A RACSIM drawing of a tree (solid) and a matching (dashed)

We will get the x-coordinates of the vertices from a particular post-order of this directed tree, that is, all children of a vertex are placed to its left, and disjoint subtrees are placed in disjoint x-intervals. We augment the input matching to a perfect matching $\mathcal{M}$ by adding dummy edges. After the algorithm is finished, these dummy edges can be safely removed to get a drawing of the input matching.

Each edge of $\mathcal{M}$ is drawn horizontally at a unique odd y-coordinate between 1 and $n-1$. The coordinates are determined as follows. In every step, our algorithm picks an edge and puts it either into the *top group*, or into the *bottom group*. Within the top group, the edges are assigned to odd rows, from top to bottom, starting from row $n-2$ if n is odd, or $n-1$ otherwise. In the bottom group, the edges are assigned also to odd rows, but from bottom to top, starting from row 1. That way, a matching edge is always assigned to an odd row between 1 and $n-1$. Once this procedure has picked an edge, the edge and its enpoints are called *placed*.

Our algorithm starts by putting the matching edge containing the root of the tree in the top group and then proceeds as follows. We partition the edges of the tree into a set of subtrees, called *ropes*, by removing the edges between two placed vertices. If there exists a subtree with three placed vertices, we call the vertex that lies on the three paths between these placed vertices a *splitter*. If the algorithm encounters a splitter, it selects the matching edge incident to the splitter, and places it next. Since a splitter is necessarily unplaced, that edge can always be picked. Otherwise, the algorithm finds an unplaced vertex that has a tree edge to a placed vertex, and adds its matching edge to the top group. This creates at most one splitter, since one of the placed vertices is adjacent to a vertex that was already placed.

It has been shown by Cabello et al. [12] that placing a vertex creates at most one splitter, and that placing a splitter does not create a new one. In every step of the algorithm, we place two vertices. The first one is either a splitter, or adjacent to a placed vertex; thus, it cannot create a new splitter. Hence, when we place a new matching edge, there will be at most one splitter in the graph.

We now describe how to determine whether to add a matching edge that contains a splitter to the top, or the bottom group. Assume that vertex v is a splitter. Since we start with drawing the root, every vertex lies in a subtree that has its local root placed. Let $T_1(v), \ldots, T_k(v)$ be the subtrees rooted in the children of v. Then, v is a splitter if and only if there are two subtrees $T_i(v)$ and $T_j(v)$ with at least one placed vertex, one of which has been placed in the last step. Without loss of generality, let $w \in T_i(v)$ be this vertex. If w was added to the top group, we add v to the bottom group; otherwise, we add it to the top group. Thus, the vertices of all trees $T_l(v)$ with $l \neq i$ will be placed on the same side (with respect to the y-coordinate) of v.

Now, we describe how to determine the x-coordinates of the vertices. To that end, for each non-leaf vertex, we will compute an order for the subtrees that are rooted in the children of this vertex. This order determines the disjoint intervals on the x-coordinates that the vertices in these subtrees will use. Let v be a vertex that is placed in an induction step. We traverse the path from v to the root of the rope it lies in. For every vertex u on this path, we determine the order of the subtrees of its children as follows. Let $T_1(u), \ldots, T_k(u)$ be the subtrees rooted in the children of u, with $v \in T_1(u)$. If v is the only placed successor of vertex u, then we assign the order $x(T_1(u)) < \ldots < x(T_k(u))$ on the x-coordinates of the subtrees; otherwise, an order has already been determined. When the algorithm is done, we know the order of the subtrees for every vertex on this path, and particularly the x-interval the subtree rooted in v lies in. We assign the highest x-coordinate of this interval to v.

Now, we show how to draw the edges. Let (u, v) be a directed edge of $\mathcal{T}$. We draw (u, v) with one slanted segment of y-length exactly 1 at vertex u, and a vertical segment at vertex v. Since we draw the edges of the matching horizontally without bends, and since the slanted segments are drawn between two consecutive horizontal grid lines, there can only be crossings between vertical segments of the tree and horizontal segments of the matching. Thus, all crossings between the tree and the matching are at right angles.

It remains to show that the drawing of the tree is itself planar. Since the drawing of the tree consists of vertical segments and slanted segments of y-length 1, any intersection has to be between a slanted and a vertical segment. Let v be a vertex of the tree. We will show that the outgoing edges of v do not induce a crossing. Since the subtree rooted in v is assigned an x-interval that contains no vertices that do not lie in its subtree, crossing can only occur between edges of this subtree. Consider the step of the algorithm in which v is placed.

First, assume that v was is not a splitter. If no successor of v has been placed so far, then all successors of v will be placed on the same side of v, with respect to the y-coordinate, and therefore not induce a crossing. Otherwise, let $T_1(v), \ldots, T_k(v)$ be the subtrees rooted in the children of v. By construction, all placed successors of v are located in the same subtree $T_1(v)$, and $T_1(v)$ is placed to the left of the other subtrees rooted in a child of v. Thus, no edge incident to v is drawn inside the x-interval assigned to $T_i(v)$, $2 \leq i \leq k$. The vertices in the other subtrees are yet to be placed, so they will all be on the same side of v, with respect to the y-coordinate, and therefore not induce a crossing.

Now, assume that v is a splitter. Then, there is a vertex u that was placed in the previous step and lies in the same rope as v. Further, there is one subtree $T_1(v)$ rooted in a child of v in which at least one vertex has been placed before u. By construction, $T_1(v)$ is placed to the left of the other subtrees rooted in a child of v. Recall that v is placed in the group opposite of u. Thus, u and all unplaced successors of v lie on the same side of v and therefore will not induce a crossing. This concludes the proof.

Finally, we will prove the area and running time bounds. We place each matching edge in a unique odd row between 1 and $(n-1)$. Thus, our drawing needs at most $(n-1)$ rows. In every column, we place exactly one vertex, so the drawing needs n columns. As for the running time, the algorithm to place the vertices clearly requires only constant running time per vertex, with the exception of traversing the tree upwards to determine the x-order of the subtrees. For that, however, we traverse every edge only once, since we stop at the first placed vertex. Thus, the running time of the algorithm is $O(n)$. $\square$

4 RACSIM Drawings with two bends per edge

In this section, we study slightly more complex classes of planar graphs, and show how to efficiently construct RACSIM drawings with two bends per edge in quadratic area. In particular, we prove that a wheel and a matching on a common set of n vertices admit a RACSIM drawing on an integer grid of size $(1.5n-1) \times (n+2)$ with two bends per edge and no bends, respectively; see Theorem 7. If the input to our problem is an outerpath—that is, an outerplanar graph whose weak dual is a path—and a matching, then a RACSIM drawing with two bends per edge and no bends, respectively, is also possible on an integer grid of size $(3n-2) \times (3n-2)$; see Theorem 8.

Theorem 7 *A wheel and a matching on a common set of n vertices admit a* RACSIM *drawing on an integer grid of size $(1.5n-1) \times (n+2)$ with two bends per edge and no bends, respectively. The drawing can be computed in $O(n)$ time.*

Proof: We denote the wheel by $\mathcal{W} = (V, E_{\mathcal{W}})$ and the matching by $\mathcal{M} = (V, E_{\mathcal{M}})$. A wheel can be decomposed into a cycle, called *rim*, a *center* vertex, and a set of edges that connect the center to the rim, called *spikes*. Let $V = \{v_1, v_2, \ldots, v_n\}$, such that v_1 is the center of $\mathcal{W}$ and $\mathcal{C} = \langle v_2, v_3, \ldots, v_n, v_2 \rangle$ is the rim of $\mathcal{W}$. Thus, $E_{\mathcal{W}} = \{(v_i, v_{i+1}) \mid i = 1, \ldots, n-1\} \cup \{(v_n, v_2)\} \cup \{(v_1, v_i) \mid i = 2, \ldots, n\}$. Let $\mathcal{M}' = (V, E_{\mathcal{M}'})$ be the matching $\mathcal{M}$ without the edge incident to v_1.

We first compute the x-coordinates of the vertices, such that $\mathcal{C} - \{(v_n, v_2)\}$ is x-monotone (if drawn with straight-line edges); see Figure 6 for an illustration. More precisely, for $i = 2, \ldots, n$ we set $x(v_i) = 2i - 3$. The y-coordinates of the vertices are computed based on the matching $\mathcal{M}'$, as follows. Let $E_{\mathcal{M}'} = \{e_1, \ldots, e_k\}$ be the matching edges with v_2 incident to e_1. For $i = 1, \ldots, k$, we assign the y-coordinate $2i - 1$ to the endpoints of e_i. Next, we assign the y-coordinate $2k + 1$ to the vertices incident to the rim without a matching edge in $\mathcal{M}'$. Finally, the center v_1 of $\mathcal{W}$ is located at point $(1, 2k + 3)$.

It remains to determine, for each edge $e \in E_{\mathcal{W}}$, where it bends, as $\mathcal{M}'$ is drawn bendless. First, let $e = (v_1, v_i)$, $i = 3, \ldots, n$ be a spike. Then, we place the bend at $(x(v_j), 2k + 2)$. Since both v_1 and v_2 are located in column 1, we can save the bend of the spike (v_1, v_2). Second, let $e = (v_i, v_{i+1})$, $i = 2, \ldots, n-1$ be an

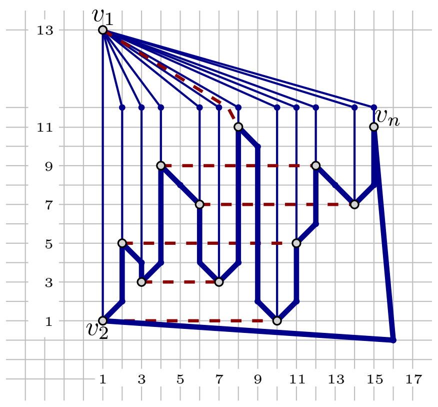


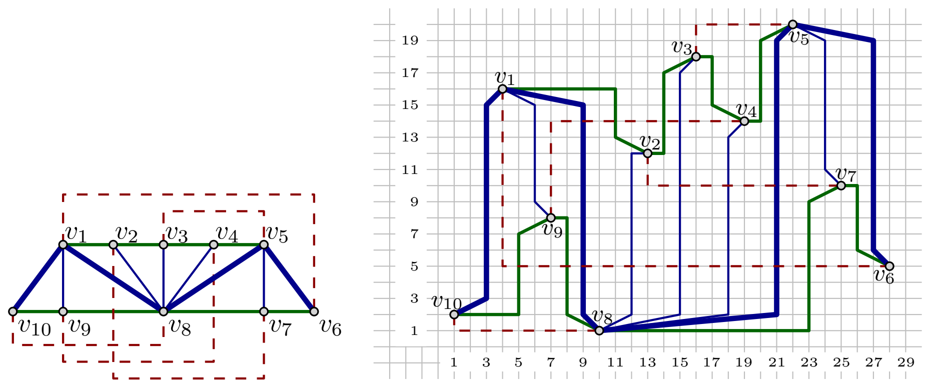
Figure 6. A RACSIM drawing of a wheel (solid; its rim is drawn bold) and a matching (dashed)

edge of the rim $\mathcal{C}$. If $y(v_{i+1}) > y(v_i)$, we place the bend at $(x(v_{i+1}), y(v_i) + 1)$. If $y(v_{i-1}) > y(v_i) > y(v_{i+1})$, we place the bend at $(x(v_{i+1}), y(v_i) - 1)$. If $y(v_i) > y(v_{i-1}), y(v_{i+1})$, the bottom port at v_i is already used. Thus, we draw the edge with two bends at $(x(v_{i+1}), y(v_i) - 1)$ and $(x(v_{i+1}), y(v_{i+1}) + 1)$. Finally, let $e = (v_n, v_2)$ be the remaining edge of the rim. Then, we place the bend at $(2n - 2, 0)$.

Our approach ensures that $\mathcal{C} - \{(v_n, v_2)\}$ is drawn x-monotone, hence planar. The last edge (v_n, v_2) of $\mathcal{C}$ is drawn outside of the bounding box containing the vertices; thus, it is crossing-free. Further, the spikes are not involved in crossings with the rim, as they are outside of the bounding box containing the rim edges. Hence, $\mathcal{W}$ is drawn planar. On the other hand, all edges of $\mathcal{M}'$ are drawn as horizontal, non-overlapping line segments. Thus, $\mathcal{M}'$ is drawn planar as well. The slanted segments of $\mathcal{W} - (v_n, v_2)$ are of y-length 1. So, they cannot be crossed by the edges of $\mathcal{M}'$. As the edge (v_n, v_2) is not involved in crossings, it follows that all crossings between $\mathcal{W}$ and $\mathcal{M}'$ form right angles.

Finally, we have to insert the matching edge (v_1, v_i) in $E_{\mathcal{M}} \setminus E'_{\mathcal{M}}$. Since v_i is not incident to a matching edge in $\mathcal{M}'$, it is placed above all matching edges. Then, $(v_1, v_i) \in \mathcal{W}$ does not cross a matching edge, so we can use this edge as a double edge.

We will now prove the area bound of the drawing algorithm. To that end, we remove all columns that contain neither a vertex, nor a bend. First, we count the rows used. Since we remove the matching edge incident to v_1, the matching $\mathcal{M}'$ has $k \leq n/2 - 1$ matching edges. We place the bottommost vertex in row 1 and the topmost vertex, that is, vertex v_1, in row $2k+3$. We add one extra bend in row 0 for the edge (v_n, v_2). Thus, our drawing uses $2k + 3 + 1 \leq n + 2$ rows. Next, we count the columns used. The vertices $v_2, \ldots, v_n$ are each placed in their own column. Every spike has exactly one bend in the column of a vertex. An edge (v_i, v_{i+1}) of rim $\mathcal{W}$ has exactly one bend in a vertex column, except for the case that $y(v_i) > y(v_{i-1}), y(v_{i+1})$, in which it needs an extra bend between v_i and v_{i+1}, $i = 1, \ldots, n-1$. Clearly, there can be at most $n/2 - 1$ vertices satisfying this condition. Since the edge (v_n, v_2) uses an extra column to the right of v_n, our drawing uses $(n - 1) + (n/2 - 1) + 1 = 1.5n - 1$ columns. □



(a) A non-planar drawing (b) A RacSim drawing

Figure 7. Two drawings of the same outerpath and matching. In both figures, the outerpath is drawn solid, the upper and the lower path are drawn bold, the spine of the spanning caterpillar is drawn very bold and the matching is drawn dashed.

Theorem 8 *An outerpath and a matching on a common set of n vertices admit a* RacSim *drawing on an integer grid of size $(3n-2) \times (2n-1)$ with two bends per edge and one bend, respectively. The drawing can be computed in $O(n \log n)$ time.*

Proof: We denote by $\mathcal{Z} = (V, E_{\mathcal{Z}})$ the outerpath and by $\mathcal{M} = (V, E_{\mathcal{M}})$ the matching. Recall that an outerpath is a biconnected outerplanar graph whose weak dual is a path of length at least two; see Figure 7a. Let $V = \{v_1, v_2, \ldots, v_n\}$ such that $\langle v_1, v_2, \ldots, v_n, v_1\rangle$ is the outerface of $\mathcal{Z}$.

We start by augmenting $\mathcal{Z}$ to a maximal outerpath $\mathcal{Z}' = (V, E_{\mathcal{Z}'})$ by triangulating its bounded faces. As $\mathcal{Z}'$ is internally-triangulated, it contains exactly two vertices of degree two, each of which belongs to a face that corresponds to an endpoint of the dual-path. Assume, without loss of generality, that $deg(v_1) = deg(v_j) = 2$ for some j with $2 < j < n$; see v_1 and v_5 in Figure 7a. We call the path $\mathcal{P}_u = (V_u, E_u) = \langle v_1, v_2, \ldots, v_j\rangle$ the *upper path* of $\mathcal{Z}'$, and the path $\mathcal{P}_\ell = (V_\ell, E_\ell) = \langle v_{j+1}, v_{j+2}, \ldots, v_n\rangle$ the *lower path* of $\mathcal{Z}'$. Observe that $V = V_u \cup V_\ell$. Further, if we remove $E_u \cup E_\ell$ from $E_{\mathcal{Z}'}$, then the resulting graph is a caterpillar $\mathcal{C}$ that spans V and whose spine alternates between vertices of V_u and V_ℓ.

We first compute the left-to-right order of the vertices of caterpillar $\mathcal{C} = (V_{\mathcal{C}}, E_{\mathcal{C}})$ as described by the algorithm supporting Theorem 4. Then, the x-coordinate of the i-th vertex in this order is $3i-2$, $i = 1, 2, \ldots, n$; see Figure 7b.

In order to compute the y-coordinates of the vertices, we first partition $\mathcal{M}$ into three matchings $\mathcal{M}_{\ell\ell} = (V_{\ell\ell}, E_{\ell\ell})$, $\mathcal{M}_{uu} = (V_{uu}, E_{uu})$ and $\mathcal{M}_{u\ell} = (V_{u\ell}, E_{u\ell})$ as follows. Let $(v, v') \in E_{\mathcal{M}}$. Then,

(i) $(v, v') \in E_{\ell\ell}$ if $v, v' \in V_\ell$,

(ii) $(v, v') \in E_{uu}$ if $v, v' \in V_u$,

(iii) $(v, v') \in E_{u\ell}$ if $v \in V_u$ and $v' \in V_\ell$.

Since $V = V_u \cup V_\ell$, it holds that $E_\mathcal{M} = E_{\ell\ell} \cup E_{uu} \cup E_{u\ell}$. In the resulting layout, the edges in $E_{\ell\ell}$ will be drawn below the edges in $E_{u\ell}$, which in turn will be drawn below the ones of E_{uu}; see Figure 7b. Thus, they will not cross each other.

Let $m_{\ell\ell} = |E_{\ell\ell}|$ and $E_{\ell\ell} = \{e_1, \ldots, e_{m_{\ell\ell}}\}$. We draw the edges from bottom to top, starting from row 1. For $i = 1, \ldots, m_{\ell\ell}$, let $e_i = (v_i, v'_i)$ with $x(v_i) < x(v'_i)$. Then, we set $y(v_i) = 4i - 1$ and $y(v'_i) = 4i - 3$. Edge e_i is drawn with a bend at $(x(v_i), y(v'_i))$. Our approach ensures that there are no crossings between edges of $\mathcal{M}_{\ell\ell}$, as they are drawn in different horizontal strips of the drawing. Similarly, we draw the edges in E_{uu} from top to bottom, starting from row $2n - 1$. Let $m_{uu} = |E_{uu}|$. By construction, the topmost vertex of $V_{\ell\ell}$ is drawn in the row $4m_{\ell\ell} - 1$, and the bottommost vertex of V_{uu} is drawn in the row $2n + 1 - 4m_{uu}$. The vertices of $V_{u\ell}$ will be drawn between these rows; see Figure 7b.

In order to draw the edges in $E_{u\ell}$, we process the vertices of the set $V_{u\ell}$ from left to right and assign y-coordinates to both endpoints of the incident matching edge. Let $m_{u\ell} = |E_{u\ell}|$ and $E_{u\ell} = \{\hat{e}_1, \ldots, \hat{e}_{m_{u\ell}}\}$. For $k = 1, \ldots, \mu$, let $\hat{e}_k = (w_k, w'_k)$ and assume without loss of generality that $x(w_k) < x(w'_k)$ and $x(w_1) < \ldots < x(w_\mu)$. We place the vertices in V_ℓ from bottom to top and the vertices in $V_u \cap V_{u\ell}$ from top to bottom. If $w_k \in V_u$, we assign the y-coordinate $4m_{\ell\ell} - 1 + 3k$ to w_k and the y-coordinate $2n + 1 - 4m_{uu}$ to w'_k; if $w_k \in V_\ell$, we switch the y-coordinates. Edge ϵ_k is drawn with a bend at $(x(w_k), y(w'_k))$. Further, every edge $\hat{e}_l \in E_{u\ell}$ with $l > k$ has its endpoints to the right of w_k and in the horizontal strip defined by the lines $y = y(w_k)$ and $y = y(w'_k)$. Hence, it will not be involved in crossings with (w_k, w'_k). This guarantees that $\mathcal{M}$ is drawn planar.

It remains to determine, for each edge $e = (v, v') \in \mathcal{Z}'$, where it bends. Without loss of generality, let e be directed from its left endpoint, say v, to its right endpoint, say v'. First, assume that $e \in E_u \cup E_\ell$ belongs to the outercycle. Then, we place its bends at $(x(v') - 2, y(v))$ and $(x(v') - 2, y(v') - \operatorname{sgn}(y(v') - y(v)))$. Second, assume that $e \in E_\mathcal{C}$ belongs to the inner caterpillar. Then, we place its bends at $(x(v') - 1, y(v) + \operatorname{sgn}(y(v) - y(v')))$ and $(x(v') - 1, y(v') - \operatorname{sgn}(y(v') - y(v)))$.

Since $\mathcal{P}_u$ and $\mathcal{P}_\ell$ are drawn x-monotone, both are drawn planar. Following similar arguments as in the proof of Theorem 4, we can show that $\mathcal{C}$ is drawn planar as well. Since $\mathcal{C}$ is drawn between $\mathcal{P}_u$ and $\mathcal{P}_\ell$, it follows that $\mathcal{Z}'$ is drawn planar, as desired. It now remains to prove that all (potential) crossings between $\mathcal{Z}'$ and $\mathcal{M}$ only involve rectilinear edge segments of $\mathcal{Z}'$, as $\mathcal{M}$ consists exclusively of rectilinear segments. As all slanted segments of $\mathcal{Z}'$ are of y-length 1, no horizontal segment of $\mathcal{M}$ can cross them. The same holds for vertical segments of $\mathcal{M}_{uu} \cup \mathcal{M}_{\ell\ell}$, as they are drawn above and below $\mathcal{P}_\ell$ and $\mathcal{P}_u$, respectively. The only possible non-rectilinear crossings are between a vertical segment of a matching edge $(u, v) \in E_{u\ell}$ and a long slanted segment of $\mathcal{C}$ incident to a spine vertex w. This crossing can only occur if w lies to the left of the vertical segment of (u, v). By construction, w is never drawn between u and v, with respect to the y-coordinate. Thus, such crossings cannot occur, which implies all crossings between $\mathcal{Z}'$ and $\mathcal{M}$ form right angles.

By the choice of the coordinates, the area requirement of our algorithm is $(3n-2) \times (2n-1)$. Since we have to sort the edges in $E_{u\ell}$ by the x-coordinates of the incident vertices, our algorithm runs in $O(n \log n)$ time. To complete the proof of this theorem, observe that the extra edges that we introduced while augmenting $\mathcal{Z}$ to $\mathcal{Z}'$ can be safely removed from the constructed layout without affecting either the crossing angles or the area of the layout. □

5 Conclusions and Open Problems

In this paper, we have studied RAC simultaneous drawings with few bends per edge. We proved that two planar graphs always admit a RAC simultaneous drawing with at most six bends per edge. For more restricted classes of graphs, we drastically improved the number of bends per edge. All of these drawings are within quadratic area. The results presented in this paper raise several questions that remain open, such as the following.

1. Is it possible to reduce the number of bends per edge for the classes of graphs that we presented in this paper?

2. What additional non-trivial classes of graphs admit a RACSIM drawing with better-than-general number of bends?

3. As a variant of the problem, it might be possible to reduce the required number of bends per edge by relaxing the strict constraint that edge intersections are at right-angles and instead ask for drawings that have close to optimal crossing resolution.

4. The computational complexity of the general problem remains open: Given two or more planar graphs on the same set of vertices and a non-negative integer k, is there a RACSIM drawing in which each graph is drawn with at most k bends per edge and the crossings are all at right angles?

5. Is it possible to achieve sub-quadratic area for special subclasses of planar graphs if we increase the number of bends?

References

[1] P. Angelini, G. D. Battista, F. Frati, V. Jelínek, J. Kratochvíl, M. Patrignani, and I. Rutter. Testing planarity of partially embedded graphs. In M. Charikar, editor, *Proc. 21st Ann. ACM-SIAM Symp. Discrete Algorithms (SODA'10)*, pages 202–221. SIAM, 2010. `doi:10.1137/1.9781611973075.19`.

[2] P. Angelini, G. D. Battista, F. Frati, M. Patrignani, and I. Rutter. Testing the simultaneous embeddability of two graphs whose intersection is a biconnected or a connected graph. *J. Discrete Algorithms*, 14:150–172, 2012. `doi:10.1007/978-3-642-19222-7_22`.

[3] P. Angelini, M. Geyer, M. Kaufmann, and D. Neuwirth. On a tree and a path with no geometric simultaneous embedding. *J. Graph Algorithms Appl.*, 16(1):37–83, 2012. `doi:10.1.1.278.6159`.

[4] E. N. Argyriou, M. A. Bekos, M. Kaufmann, and A. Symvonis. Geometric RAC simultaneous drawings of graphs. *J. Graph Algorithms Appl.*, 17(1):11–34, 2013. doi:10.7155/jgaa.00282.

[5] M. A. Bekos, M. Gronemann, and C. N. Raftopoulou. Two-page book embeddings of 4-planar graphs. In E. W. Mayr and N. Portier, editors, *31st Int. Symp. Theor. Aspects Comput. Sci. (STACS'14)*, volume 25 of *LIPIcs*, pages 137–148. Schloss Dagstuhl - Leibniz-Zentrum fuer Informatik, 2014. doi:10.4230/LIPIcs.STACS.2014.137.

[6] M. A. Bekos, T. C. van Dijk, P. Kindermann, and A. Wolff. Simultaneous drawing of planar graphs with right-angle crossings and few bends. In M. S. Rahman and E. Tojima, editors, *Proc. 9th Int. Workshop on Algorithms and Computation (WALCOM'15)*, Lecture Notes Comput. Sci. Springer-Verlag, Feb. 2015. To appear.

[7] F. Bernhart and P. C. Kainen. The book thickness of a graph. *J. Comb. Theory, Series B*, 27(3):320–331, 1979. doi:10.1016/0095-8956(79)90021-2.

[8] T. Bläsius, S. G. Kobourov, and I. Rutter. Simultaneous embedding of planar graphs. In R. Tamassia, editor, *Handbook of Graph Drawing and Visualization*, chapter 11, pages 349–381. CRC Press, 2013. URL: http://cs.brown.edu/~rt/gdhandbook/.

[9] T. Blsius, A. Karrer, and I. Rutter. Simultaneous embedding: Edge orderings, relative positions, cutvertices. In S. Wismath and A. Wolff, editors, *Proc. 21st Int. Symp. Graph Drawing*, volume 8242 of *Lecture Notes Comput. Sci.*, pages 220–231. Springer-Verlag, 2013. doi:10.1007/978-3-319-03841-4_20.

[10] P. Brass, E. Cenek, C. A. Duncan, A. Efrat, C. Erten, D. P. Ismailescu, S. G. Kobourov, A. Lubiw, and J. S. Mitchell. On simultaneous planar graph embeddings. *Comput. Geom. Theory Appl.*, 36(2):117–130, 2007. doi:10.1016/j.comgeo.2006.05.006.

[11] G. Brightwell and E. R. Scheinerman. Representations of planar graphs. *SIAM J. Discrete Math.*, 6(2):214–229, 1993. doi:10.1137/0406017.

[12] S. Cabello, M. van Kreveld, G. Liotta, H. Meijer, B. Speckmann, and K. Verbeek. Geometric simultaneous embeddings of a graph and a matching. *J. Graph Algorithms Appl.*, 15(1):79–96, 2011. doi:10.7155/jgaa.00218.

[13] G. Cornuéjols, D. Naddef, and W. Pulleyblank. Halin graphs and the travelling salesman problem. *Math. Program.*, 26(3):287–294, 1983. doi:10.1007/BF02591867.

[14] C. Erten and S. G. Kobourov. Simultaneous embedding of a planar graph and its dual on the grid. *Theory Comput. Syst.*, 38(3):313–327, 2005. doi:10.1007/3-540-36136-7_50.

[15] C. Erten and S. G. Kobourov. Simultaneous embedding of planar graphs with few bends. *J. Graph Algorithms Appl.*, 9(3):347–364, 2005. `doi:10.7155/jgaa.00113`.

[16] A. Estrella-Balderrama, E. Gassner, M. Jnger, M. Percan, M. Schaefer, and M. Schulz. Simultaneous geometric graph embeddings. In S.-H. Hong, T. Nishizeki, and W. Quan, editors, *Proc. 16th Int. Symp. Graph Drawing*, volume 4875 of *Lecture Notes Comput. Sci.*, pages 280–290. Springer-Verlag, 2008. `doi:10.1007/978-3-540-77537-9_28`.

[17] S. Felsner. *Geometric Graphs and Arrangements*. Vieweg Verlag, 2004. `doi:10.1007/978-3-322-80303-0`.

[18] L. Grilli, S.-H. Hong, J. Kratochvl, and I. Rutter. Drawing simultaneously embedded graphs with few bends. In C. Duncan and A. Symvonis, editors, *Proc. 22nd Int. Symp. Graph Drawing (GD'14)*, volume 8871 of *Lecture Notes Comput. Sci.*, pages 40–51. Springer-Verlag, 2014. `doi:10.1007/978-3-662-45803-7_4`.

[19] B. Haeupler, K. R. Jampani, and A. Lubiw. Testing simultaneous planarity when the common graph is 2-connected. *J. Graph Algorithms Appl.*, 17(3):147–171, 2013. `doi:10.7155/jgaa.00289`.

[20] C. Haslinger and P. F. Stadler. Rna structures with pseudo-knots: Graph-theoretical, combinatorial, and statistical properties. *Bull. Math. Biol.*, 61(3):437–467, 1999. `doi:10.1006/bulm.1998.0085`.

[21] L. Heath. Embedding outerplanar graphs in small books. *SIAM J. Algebraic and Discrete Methods*, 8(2):198–218, 1987. `doi:10.1137/0608018`.

[22] L. S. Heath. *Algorithms for Embedding Graphs in Books*. PhD thesis, University of North Carolina, Chapel Hill, 1985. URL: `http://www.cs.unc.edu/techreports/85-028.pdf`.

[23] P. C. Kainen and S. Overbay. Extension of a theorem of whitney. *Appl. Math. Lett.*, 20(7):835–837, 2007. `doi:10.1016/j.aml.2006.08.019`.

[24] F. Kammer. Simultaneous embedding with two bends per edge in polynomial area. In L. Arge and R. Freivalds, editors, *Proc. 10th Scand. Workshop on Algorithm Theory (SWAT'06)*, volume 4059 of *Lecture Notes Comput. Sci.*, pages 255–267. Springer-Verlag, 2006. `doi:10.1007/11785293_25`.

[25] M. Kaufmann and R. Wiese. Embedding vertices at points: Few bends suffice for planar graphs. *J. Graph Algorithms Appl.*, 6(1):115–129, 2002. `doi:10.7155/jgaa.00046`.

[26] C. S. J. A. Nash-Williams. Decomposition of finite graphs into forests. *J. London Math. Soc.*, 1(1):12–12, 1964. `doi:10.1016/0012-365X(91)90377-E`.

[27] T. Nishizeki and N. Chiba. *Planar Graphs: Theory and Algorithms*, chapter Chapter 10. Hamiltonian Cycles, pages 171–184. Dover Books Math. Courier Dover Pub., 2008. URL: `http://store.doverpublications.com/048646671x.html`.

[28] M. Schaefer. Toward a theory of planarity: Hanani-tutte and planarity variants. *J. Graph Algorithms Appl.*, 17(4):367–440, 2013. `doi:10.7155/jgaa.00298`.

[29] A. Wigderson. The complexity of the hamiltonian circuit problem for maximal planar graphs. Technical Report TR-298, EECS Department, Princeton University, 1982. URL: `http://www.math.ias.edu/~avi/PUBLICATIONS/MYPAPERS/W82a/tech298.pdf`.