1-Bend RAC Drawings of NIC-Planar Graphs in Quadratic Area

Steven Chaplick, Fabian Lipp, Alexander Wolff, and Johannes Zink

Lehrstuhl für Informatik I, Universität Würzburg.
http://www1.informatik.uni-wuerzburg.de/en/staff/

Abstract

A drawing of a graph is called *1-planar* if every edge is crossed at most once. A 1-planar drawing is called *independent-crossing planar* (*IC-planar*) if no two pairs of crossing edges share a vertex. A 1-planar drawing is called *near-independent-crossing planar* (*NIC-planar*) if any two pairs of crossing edges share at most one vertex. The 1-planar, NIC-planar, and IC-planar graphs are the graphs that admit a 1-planar, NIC-planar, and IC-planar drawing, respectively. The NIC-planar graphs are a subset of the 1-planar graphs and a superset of the IC-planar graphs, which are important *beyond-planar* graph classes. We constructively show that every n-vertex NIC-plane graph admits a NIC-planar drawing with only right-angle crossings (RAC) and at most one bend per edge on a grid of size $O(n) \times O(n)$. Our construction takes linear time. We also give an overview of the relationships between several classes of 1-planar and RAC graphs.

1 Introduction

In graph theory and graph drawing, *beyond-planar* graph classes have experienced increasing interest in recent years. A prominent example is the class of *1-planar graphs*, that is, graphs that admit a drawing where each edge is crossed at most once. 1-planar graphs were introduced by Ringel [13] in 1965; Kobourov et al. [11] surveyed them recently. Another example that has received considerable attention are *RAC$_k$ graphs*, that is, graphs that admit a poly-line drawing where all crossings are at right angles and each edge has at most k bends. RAC$_k$ graphs were introduced by Didimo et al. [7]. We investigate the relationships between (certain subclasses of) 1-planar graphs and RAC$_k$ graphs that admit drawings on a polynomial-size grid. Known results and our contributions are summarized in Fig. 1.

Basic Terminology. A mapping Γ is called a *drawing* of the graph $G = (V, E)$ if each vertex $v \in V$ is mapped to a point in $\mathbb{R}^2$ and each edge $\{u, v\}$ is mapped to a simple open Jordan curve in $\mathbb{R}^2$ such that the endpoints of this curve are $\Gamma(u)$ and $\Gamma(v)$. For convenience, we will refer to the points and simple open Jordan curves of a drawing as vertices and edges. The topologically connected regions of $\mathbb{R}^2 \setminus \Gamma$ are called *faces* of Γ. The unbounded face of Γ is called *outer* face; the other faces are *inner* faces. An equivalence class of drawings of a graph G having the same set of faces and the same outer face is called an *embedding* of G.

A *k-bend (poly-line) drawing* is a drawing in which every edge is drawn as a connected sequence of at most $k + 1$ line segments. The up to k inner vertices of an edge connecting these line segments are called *bend points* or *bends*. A 0-bend drawing is called *straight-line*. A *drawing on the grid of size $w \times h$* is a drawing where every vertex, bend point, and crossing point has integer coordinates in the range $[0, w] \times [0, h]$. Recall that a drawing is *1-planar* if every edge is crossed at most once. A 1-planar drawing is called *independent-crossing planar* (*IC-planar*) if no two pairs of crossing edges share a vertex. A 1-planar drawing is called *near-independent-crossing planar* (*NIC-planar*) if any two pairs of crossing edges share at most one vertex. A drawing is called *right-angle-crossing* (*RAC*) if it is a poly-line drawing and for each crossing point c, there are at most two edges that cross in c, there is no bend point in c, and the line segments of the edges that cross in c intersect in a right angle. As

34th European Workshop on Computational Geometry, Berlin, Germany, March 21–23, 2018.
This is an extended abstract of a presentation given at EuroCG'18. It has been made public for the benefit of the community and should be considered a preprint rather than a formally reviewed paper. Thus, this work is expected to appear eventually in more final form at a conference with formal proceedings and/or in a journal.

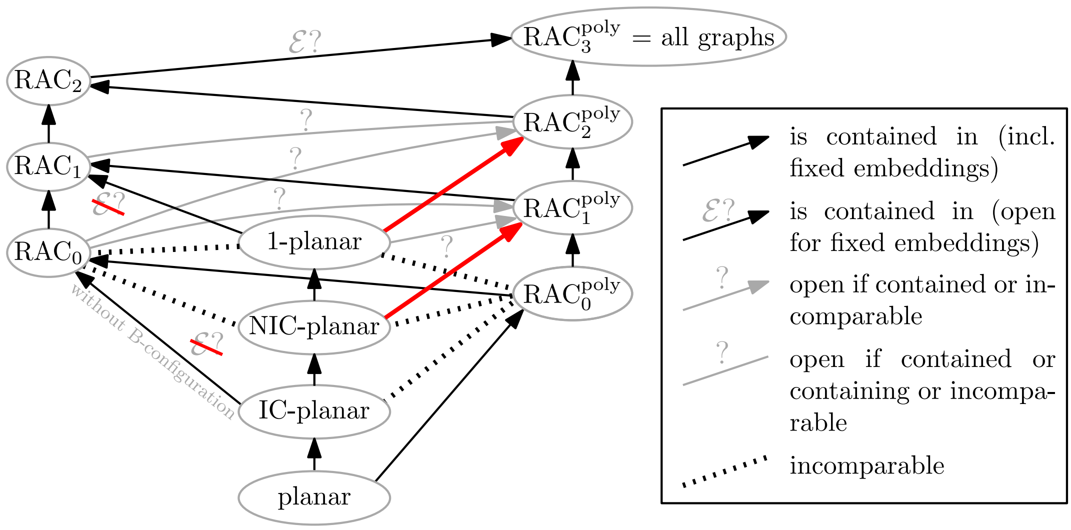


Figure 1 Relationships between (beyond-)planar graphs and RAC graphs. Our main results are the containment relationships indicated by the thick red arrows.

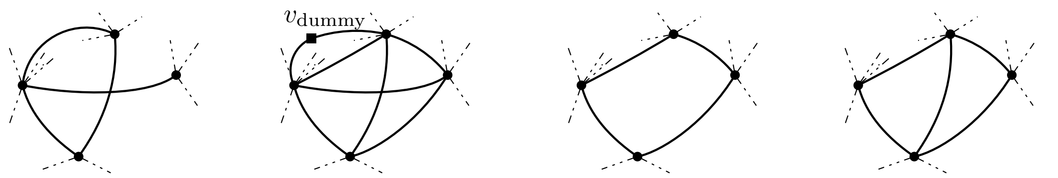
mentioned above, a drawing is called RAC$_k$ if it is RAC and k-bend. The *planar*, *1-planar*, *NIC-planar*, *IC-planar*, and *RAC$_k$* graphs are the graphs that admit a crossing-free, 1-planar, NIC-planar, IC-planar, and RAC$_k$ drawing, respectively. More specifically, RAC$_k^{\text{poly}}$ is the set of graphs that admit a RAC$_k$ drawing on a grid of size polynomial in the number of vertices. A *plane*, *1-plane*, *NIC-plane*, and *IC-plane* graph is a graph given with a specific planar, 1-planar, NIC-planar, and IC-planar embedding, respectively.

Previous Work. In the diagram in Fig. 1, we give an overview of the relationships between classes of 1-planar graphs and RAC$_k$ graphs. Clearly, the planar graphs are a subset of the IC-planar graphs, which are a subset of the NIC-planar graphs, which are a subset of the 1-planar graphs. It is well known that every plane graph can be drawn with straight-line edges on a grid of quadratic size [6, 14]. Every IC-planar graph admits an IC-planar RAC$_0$ drawing but not always in polynomial area [4]. Moreover, there are graphs in RAC$_0^{\text{poly}}$ that are not 1-planar [8] and, therefore, also not IC-planar. The class of RAC$_0$ graphs is incomparable with the classes of NIC-planar graphs [1] and 1-planar graphs [8]. Bekos et al. [2] showed that every 1-planar graph admits a 1-planar 1-bend RAC drawing, but their recursive drawings may need exponential area. Brandenburg [3] claimed that every 1-planar graph admits a 1-planar 1-bend RAC drawing where the vertices lie on a polynomial-size grid. In a private communication, he later retracted his claim—therefore, this question remains open. Every graph admits a RAC$_3$ drawing in polynomial area [7]. This does not hold if a given embedding of a planarization of the graph must be preserved [7].

Our Contributions. Our main result is as follows.

▶ **Theorem 1.** *Every n-vertex NIC-planar graph G admits a NIC-planar 1-bend RAC drawing on a grid of size* $(16n - 32) \times (8n - 16)$. *Given a NIC-planar embedding $\mathcal{E}$ of G, a drawing that has these characteristics and respects $\mathcal{E}$ can be computed in $O(n)$ time.*

For IC-plane graphs, this reduces the number of bends compared to a recent result of Liotta and Montecchiani [12] who showed that every IC-planar graph admits an IC-planar RAC$_2$ drawing on a grid of quadratic size. We have also shown (see Zink's master's thesis [15]) that every 1-plane graph admits a 1-planar RAC$_2$ drawing in polynomial area and, by a small



(a) crossing as it initially appears **(b)** empty kite and subdivided original edge **(c)** empty quadrangle **(d)** divided quadrangle

Figure 2 Modifications of crossings and computation of the biconnected canonical ordering.

modification of the algorithm by Bekos et al. [2], that not only every 1-planar, but even every 1-plane graph admits a 1-planar RAC_1 drawing. We can also show, by a small modification of the algorithm by Brandenburg et al. [4], that not only every IC-planar, but even every IC-plane graph without so called *B-configurations* admits a IC-planar RAC_0 drawing. Due to space considerations, we omit these results here.

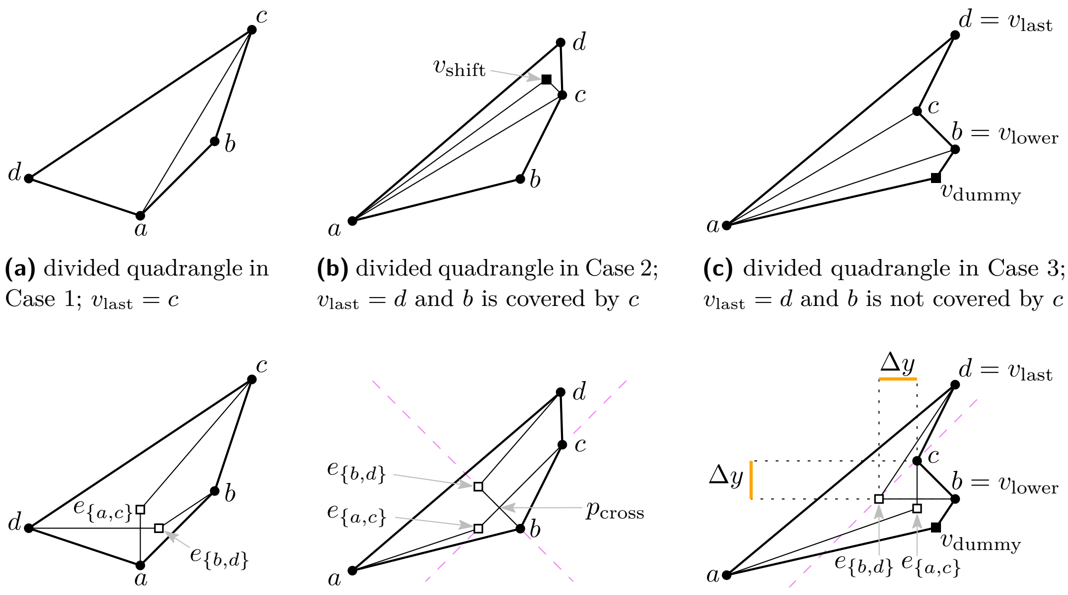
2 1-Bend RAC Drawings of NIC-Plane Graphs in Quadratic Area

Our algorithm takes an n-vertex NIC-plane graph $(G, \mathcal{E})$ as input and returns a NIC-planar RAC_1 drawing of G on a grid of size $O(n) \times O(n)$ while maintaining $\mathcal{E}$. We now describe the algorithm. We omit a formal correctness proof due to lack of space.

Preprocessing. We first aim to make the NIC-plane input graph biconnected and planar so that we can draw it using the algorithm by Harel and Sardas [9]. Around each crossing in $\mathcal{E}$, we insert up to four dummy edges to obtain *empty kites*. A *kite* is a K_4 that is embedded such that (i) every vertex lies on the boundary of the outer face, (ii) there is exactly one crossing, and (iii) this crossing doesn't lie on the boundary of the outer face. A kite K as a subgraph of a graph H is said to be *empty* if there is no edge of $H \backslash K$ that is on an inner face of K or crosses edges of K. Whenever we insert a dummy edge, we may create a pair of parallel edges. Then, we subdivide the original edge participating in this pair by a dummy vertex (see the transition from Fig. 2a to 2b). Note that we never create parallel dummy edges since G is NIC-planar. After this, we remove both crossing edges from each empty kite and obtain *empty quadrangles* (see Fig. 2c). We store each such empty quadrangle in a list Q. At the end of the preprocessing, we make the resulting plane graph biconnected via, e.g., the algorithm of Hopcroft and Tarjan [10]. Let $(G', \mathcal{E}')$ be the resulting plane biconnected graph.

Drawing Step. Now, we draw a graph that we obtain from $(G', \mathcal{E}')$. We use the algorithm by Harel and Sardas [9], which is a generalization of the algorithm of Chrobak and Payne [5], which in turn is based on the shift algorithm of de Fraysseix et al. [6]. The algorithm of Harel and Sardas consists of two phases. Given a plane biconnected graph H, in the first phase a biconnected canonical ordering Π of the vertices in the plane input graph is computed. In the second phase, H is drawn according to Π on a grid of size $(2|V(H)| - 4) \times (|V(H)| - 2)$. Biconnected canonical orderings are a generalization of canonical orderings that assume only biconnectivity (instead of triconnectivity). Unlike the classical shift algorithm, the algorithm of Harel and Sardas computes the (biconnected) canonical ordering *bottom-up*, which we will exploit here. Let $\Pi_k = (v_1, \ldots, v_k)$ be a partial biconnected canonical ordering of H after step k, and let H_k be the plane subgraph of H induced by Π_k. We say that a vertex u is *covered* by v_k if u is on the boundary of the outer face of H_{k-1}, but not on that of H_k.

We perform the following additional operations when we compute the biconnected canonical ordering. Whenever we reach an empty quadrangle $q = (a, b, c, d)$ in the list Q for



(a) divided quadrangle in Case 1; $v_{\text{last}} = c$

(b) divided quadrangle in Case 2; $v_{\text{last}} = d$ and b is covered by c

(c) divided quadrangle in Case 3; $v_{\text{last}} = d$ and b is not covered by c

(d) Case 1 after inserting the crossing edges

(e) Case 2 after inserting the crossing edges

(f) Case 3 after inserting the crossing edges

Figure 3 The three cases of the drawing step (a)–(c) and the reinsertion step (d)–(f) in our algorithm. For orientation, lines with slope 1 or -1 are dashed violet.

the first time, i.e., when the first vertex of q—say a—is added to the biconnected canonical ordering, we insert an edge inside q from a to the vertex opposite a in q, that is, to c. We call the resulting structure a *divided quadrangle* (see Fig. 2d). In two special cases, we perform further modifications of the graph. They will help us to guarantee a correct reinsertion of the crossing edges in the next step of the algorithm. Namely, when we encounter the last vertex $v_{\text{last}} \in \{b, c, d\}$ of q, we distinguish three cases.

Case 1: $v_{\text{last}} = c$ (see Fig. 3a).

In this case, we perform no extra operation.

Case 2: $v_{\text{last}} \in \{b, d\}$, and the other of these two vertices is covered by c (see Fig. 3b).

We insert a dummy vertex v_{shift}, which we call *shift vertex*, into the current biconnected canonical ordering directly before v_{last} and make it adjacent to a and c. Later, we will remove v_{shift}, but for now it forces the algorithm of Harel and Sardas to shift a and c away from each other before v_{last} is added.

Case 3: $v_{\text{last}} \in \{b, d\}$, and neither b nor d is covered by c (see Fig. 3c).

Let $\{v_{\text{lower}}\} = \{b, d\} \setminus v_{\text{last}}$. We subdivide the edge $\{a, v_{\text{lower}}\}$ via a dummy vertex v_{dummy}. If $\{a, v_{\text{lower}}\}$ is an original edge of the input graph, this edge will be bent at v_{dummy} in the final drawing. We insert v_{dummy} into the current biconnected canonical ordering directly before v_{lower}. To obtain a divided quadrangle again, we insert the dummy edge $\{a, v_{\text{lower}}\}$, which we will remove before we reinsert the crossing edges. This will give us some extra space inside the triangle $(a, v_{\text{dummy}}, v_{\text{lower}})$ for a bend point.

We draw the resulting plane biconnected $\hat{n}$-vertex graph $(\hat{G}, \hat{\mathcal{E}})$ according to its biconnected canonical ordering $\hat{\Pi}$ via the algorithm by Harel and Sardas and obtain a crossing-free drawing $\hat{\Gamma}$. We do not modify the actual drawing phase.

Postprocessing (Reinserting the Crossing Edges). We refine the underlying grid of $\hat{\Gamma}$ by a factor of 2 in both dimensions. Let $q = (a, b, c, d)$ be a quadrangle in Q, where a is the first and v_{last} the last vertex in $\hat{\Pi}$ among the vertices in q. From q, we first remove the chord edge $\{a, c\}$ and obtain an empty quadrangle. Then, we distinguish three cases for reinserting the crossing edges that we removed in the preprocessing. These are the same cases as in the description of the modified computation of the biconnected canonical ordering above.

Case 1: $v_{\text{last}} = c$ (see Fig. 3a).

Since c is adjacent to a, b, and d in $\hat{G}$, it has the largest y-coordinate among the vertices in q. Assume that $y(d)$ is smaller or equal to $y(b)$ since the other case is symmetric. An example of a quadrangle in this case before and after the reinsertion of the crossing edges is given in Figs. 3a and 3d, respectively. We will have a crossing point at $(x(a), y(d))$. To this end, we insert the edge $\{a, c\}$ with a bend at $e_{\{a,c\}} = (x(a), y(d) + 1)$ and we insert the edge $\{b, d\}$ with a bend at $e_{\{b,d\}} = (x(a) + 1, y(d))$.

Case 2: $v_{\text{last}} \in \{b, d\}$, and the other of these two vertices is covered by c (see Fig. 3b).

Assume that $y(d) > y(b)$; the other case is symmetric. An example of a quadrangle in this case before and after the reinsertion of the crossing edges is given in Figs. 3b and 3e, respectively. Here, we remove v_{shift} in addition to removing the edge $\{a, c\}$. We define the crossing point $p_{\text{cross}} = (x_{\text{cross}}, y_{\text{cross}})$ as the intersection point of the lines with slope 1 and -1 through c and b, respectively. The coordinates of this crossing point are $x_{\text{cross}} = (x(c) - y(c) + x(b) + y(b))/2$ and $y_{\text{cross}} = (-x(c) + y(c) + x(b) + y(b))/2$. Despite the fact that both coordinates are the result of a division by 2, both are integers—recall that we refined the grid by a factor of 2 in each dimension. We place the two bend points onto the same lines at the closest grid points that are next to p_{cross}. In other words, we draw the edge $\{a, c\}$ with a bend point at $e_{\{a,c\}} = (x_{\text{cross}} - 1, y_{\text{cross}} - 1)$ and we insert the edge $\{b, d\}$ with a bend point at $e_{\{b,d\}} = (x_{\text{cross}} - 1, y_{\text{cross}} + 1)$. We do not intersect or touch the edge $\{a, d\}$ because we shifted a far enough away from c by the extra shift due to v_{shift}. Moreover, the points $e_{\{a,c\}}$ and p_{cross} on the line with slope 1 through c are inside the empty quadrangle q since b is covered by c (then b is below the line with slope 1 through c) and $y(b)$ is at most equal to $y(e_{\{a,c\}})$.

Case 3: $v_{\text{last}} \in \{b, d\}$, and neither b nor d is covered by c (see Fig. 3c).

Assume that $y(d) > y(b)$; again, the other case is symmetric. An example of a quadrangle in this case before and after the reinsertion of the crossing edges is given in Figs. 3c and 3f, respectively. Note that the edge $\{a, b\}$ is the dummy edge which we inserted during the computation of $\hat{\Pi}$ and next to this edge, there is the path $(a - v_{\text{dummy}} - b)$. This path is the former edge $\{a, b\}$. We will reinsert the edges $\{a, c\}$ and $\{b, d\}$ such that they cross in $(x(c), y(b))$. We will bend the edge $\{b, d\}$ on the line with slope 1 through c at $y = y(b)$ because from this point we always "see" d inside q. So, we define $x_{\text{bend}} := x(c) - \Delta y$ with $\Delta y := y(c) - y(b)$. First, we remove the dummy edge $\{a, b\}$. Second, we insert the edge $\{a, c\}$ with a bend point at $e_{\{a,c\}} = (x(c), y(b) - 1)$. Third, we insert the edge $\{b, d\}$ with a bend point at $e_{\{b,d\}} = (x_{\text{bend}}, y(b))$. Note that $e_{\{a,c\}}$ might be below the straight line segment $\overline{ab}$ since a could have been shifted away from c several times. However, $e_{\{a,c\}}$ cannot be on or below the path $(a - v_{\text{dummy}} - b)$ because $y(v_{\text{dummy}}) < y(e_{\{a,c\}})$ and the slope of the line segment $\overline{v_{\text{dummy}}b}$ is either greater than 1 or negative. Therefore, the crossing edges $\{a, c\}$ and $\{b, d\}$ lie completely inside the pentagon $(a, v_{\text{dummy}}, b, c, d)$.

After we have reinserted the crossing edges into each quadrangle of Q, we remove all dummy edges and transform the remaining dummy vertices to bend points. The resulting drawing Γ is a RAC_1 drawing that preserves the embedding of the NIC-plane input graph $(G, \mathcal{E})$. Our algorithm runs in linear time. Since the shift algorithm draws $\hat{\Gamma}$ on a grid of size $(2\hat{n} - 4) \times (\hat{n} - 2)$, which we refined by a factor of 2, and $\hat{n} \leq 4n - 6$, Γ lies on a grid of size at most $(16n - 32) \times (8n - 16)$.

3 Conclusion and Open Questions

We have presented an algorithm for drawing any NIC-plane graph on a small grid with right-angle crossings and at most one bend per edge. Our algorithm is based on the shift algorithm for 2-connected graphs by Harel and Sardas [9]. Before and while we execute their algorithm, we modify the graph (incl. removing the crossing edges) to obtain faces with nice properties into which we reinsert the crossing edges afterwards.

The diagram in Fig. 1 leaves some open questions. Does every 1-planar graph admit a 1-planar 1-bend RAC drawing in polynomial area? Can every graph in RAC_0 be drawn in polynomial area if we allow one or two bends per edge? What is the relationship between RAC_1 and $\mathrm{RAC}_2^{\mathrm{poly}}$?

References

1. Christian Bachmaier, Franz J. Brandenburg, Kathrin Hanauer, Daniel Neuwirth, and Josef Reislhuber. NIC-planar graphs. *Discrete Appl. Math.*, 232:23–40, 2017. doi:10.1016/j.dam.2017.08.015.
2. 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.
3. Franz J. Brandenburg. Compact 1-bend RAC drawings of 1-planar graphs. In *Proc. 33rd Europ. Workshop Comput. Geom. (EuroCG'17)*, pages 129–132, 2017. URL: http://csconferences.mah.se/eurocg2017/proceedings.pdf.
4. Franz J. Brandenburg, Walter Didimo, William S. Evans, Philipp Kindermann, Giuseppe Liotta, and Fabrizio Montecchiani. Recognizing and drawing IC-planar graphs. *Theor. Comput. Sci.*, 636:1–16, 2016. doi:10.1016/j.tcs.2016.04.026.
5. 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.
6. Hubert de Fraysseix, János Pach, and Richard Pollack. How to draw a planar graph on a grid. *Combinatorica*, 10(1):41–51, 1990. doi:10.1007/BF02122694.
7. 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.
8. Peter Eades and Giuseppe Liotta. Right angle crossing graphs and 1-planarity. *Discrete Appl. Math.*, 161(7-8):961–969, 2013. doi:10.1016/j.dam.2012.11.019.
9. David Harel and Meir Sardas. An algorithm for straight-line drawing of planar graphs. *Algorithmica*, 20(2):119–135, 1998. doi:10.1007/PL00009189.
10. John E. Hopcroft and Robert Endre Tarjan. Efficient algorithms for graph manipulation (algorithm 447). *Commun. ACM*, 16(6):372–378, 1973. doi:10.1145/362248.362272.
11. Stephen G. Kobourov, Giuseppe Liotta, and Fabrizio Montecchiani. An annotated bibliography on 1-planarity. arXiv report, 2017. URL: http://arxiv.org/abs/1703.02261.
12. Giuseppe Liotta and Fabrizio Montecchiani. L-visibility drawings of IC-planar graphs. *Inf. Process. Lett.*, 116(3):217–222, 2016. doi:10.1016/j.ipl.2015.11.011.
13. Gerhard Ringel. Ein Sechsfarbenproblem auf der Kugel. *Abhandlungen aus dem Mathematischen Seminar der Universität Hamburg*, 29(1–2):107–117, 1965.
14. Walter Schnyder. Embedding planar graphs on the grid. In *Proc. 1st ACM-SIAM Symp. Discrete Algorithms (SODA'90)*, pages 138–148, 1990. URL: http://dl.acm.org/citation.cfm?id=320176.
15. Johannes Zink. 1-planar RAC drawings with bends. Master's thesis, Institut für Informatik, Universität Würzburg, 2017. Under Review. URL: http://www1.pub.informatik.uni-wuerzburg.de/pub/theses/2017-zink-master.pdf.