

# Tangling and Untangling Trees on Point-Sets

Giuseppe Di Battista  

Roma Tre University, Italy

Giuseppe Liotta  

University of Perugia, Italy

Maurizio Patrignani  

Roma Tre University, Italy

Antonios Symvonis  

School of Applied Mathematical & Physical Sciences, National Technical University of Athens, Greece

Ioannis G. Tollis  

Department of Computer Science, University of Crete, Heraklion, Greece

---

## Abstract

We study a question that lies at the intersection of classical research subjects in Topological Graph Theory and Graph Drawing: Computing a drawing of a graph with a prescribed number of crossings on a given set  $S$  of points, while ensuring that its curve complexity (i.e., maximum number of bends per edge) is bounded by a constant. We focus on trees: Let  $T$  be a tree,  $\vartheta(T)$  be its thrackle number, and  $\chi$  be any integer in the interval  $[0, \vartheta(T)]$ . In the *tangling* phase we compute a topological linear embedding of  $T$  with  $\vartheta(T)$  edge crossings and a constant number of spine traversals. In the *untangling* phase we remove edge crossings without increasing the spine traversals until we reach  $\chi$  crossings. The computed linear embedding is used to construct a drawing of  $T$  on  $S$  with  $\chi$  crossings and constant curve complexity. Our approach gives rise to an  $O(n^2)$ -time algorithm for general trees and an  $O(n \log n)$ -time algorithm for paths. We also adapt the approach to compute RAC drawings, i.e. drawings where the angles formed at edge crossings are  $\frac{\pi}{2}$ .

**2012 ACM Subject Classification** Mathematics of computing  $\rightarrow$  Graph algorithms; Mathematics of computing  $\rightarrow$  Graph theory

**Keywords and phrases** Tree drawings, Prescribed edge crossings, Thrackle, Curve complexity, Point-set embeddings, RAC drawings, Topological linear embeddings

**Digital Object Identifier** 10.4230/LIPIcs.GD.2025.8

**Related Version** Full Version: <https://arxiv.org/abs/2508.18535> [5]

**Funding** *Giuseppe Di Battista*: Research supported by MUR PRIN Project no. 2022TS4Y3N.

*Giuseppe Liotta*: Research supported by MUR PON Proj. ARS01 00540 and by MUR PRIN Project no. 2022TS4Y3N.

*Maurizio Patrignani*: Research supported by MUR PRIN Project no. 2022TS4Y3N.

*Antonios Symvonis*: Supported by HFRI Grant No: 26320.

**Acknowledgements** This work started at the Bertinoro Workshop on Graph Drawing BWGD 2024.

## 1 Introduction

In this paper we study the following problem: Given an  $n$ -vertex tree  $T$ , a set  $S$  of  $n$  distinct points in the plane, and a non-negative integer  $\chi$ , find a simple drawing  $\Gamma$  of  $T$  with the following properties:

1.  $\Gamma$  is a *point-set embedding* of  $T$  on  $S$ , i.e. its vertices are the points of  $S$ ,



© Giuseppe Di Battista, Giuseppe Liotta, Maurizio Patrignani, Antonios Symvonis, and Ioannis G. Tollis;

licensed under Creative Commons License CC-BY 4.0

33rd International Symposium on Graph Drawing and Network Visualization (GD 2025).

Editors: Vida Dujmović and Fabrizio Montecchiani; Article No. 8; pp. 8:1–8:18



Leibniz International Proceedings in Informatics

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

2. the edges of  $T$  are represented by polylines with constant *curve complexity*, measured as the maximum number of bends per edge, and
3. the edges cross exactly  $\chi$  times in  $\Gamma$ .

In the following we refer to them as Property 1, Property 2, and Property 3.

The problem locates itself at the intersection of three well studied research subjects in Graph Drawing and Graph Theory, namely the study of point-set embeddings, the study of drawings with limited curve complexity, and the study of drawings with prescribed numbers of edge crossings. Before describing our contribution, we briefly recall each such topic.

Property 1 – For a graph  $G$  with  $n$  vertices and a set  $S$  of  $n$  distinct points in the plane, a *point-set embedding* of  $G$  on  $S$  is a drawing of  $G$  where each vertex is mapped to a distinct point of  $S$ . Most of the literature on this topic is about planar ( $\chi = 0$ ) point-set embeddings of graphs (see, e.g., [7, 8, 9, 10, 11, 13, 35, 43, 44]). Specifically, it is NP-complete to test if a planar graph  $G$  has a straight-line planar point-set embedding on a given point-set [13]. If  $G$  is an outerplanar graph and the points are in general position such a drawing can be constructed in polynomial time (see, e.g., [8, 9, 10, 11]).

Property 2 – Together with the edge crossing minimization, the minimization of the edge bends and/or of the bends per edge are among the oldest optimization questions in graph drawing (see, e.g., [15, 38, 50]). In particular, the relationship between point-set embeddings and curve complexity has been studied when the mapping between the vertices of the graph and the points is specified as part of the input as well as when it is partially specified or not specified (see, e.g., [3, 29, 30, 31, 32, 33, 34, 39, 45, 46]). Notably, Kaufmann and Wiese proved that when the mapping is not specified a planar graph admits a crossing-free point-set embedding with constant curve complexity, namely at most two bends per edge [39, 40].

Property 3 – The *thrackle bound*  $\vartheta(G)$  of a graph  $G = (V, E)$  with  $m = |E|$  edges is the number of crossings that a drawing of  $G$  would have if every edge crosses every other non-adjacent edge exactly once. It is well known that  $\vartheta(G) = (m(m+1) - \sum_{v \in V} \deg^2(v))/2$ ; see, e.g., [48, 49]. As an example, for a cycle  $C_n$  with  $n$  vertices the above formula gives  $\vartheta(C_n) = n(n-3)/2$ . For a path  $P_n$  with  $n$  vertices, where all but two vertices have degree two, the formula gives  $\vartheta(P_n) = (n-2)(n-3)/2$ . A *thrackle* is a drawing of a graph  $G$  with exactly  $\vartheta(G)$  crossings. Not all graphs admit thrackles. For example, a cycle  $C_4$  with 4 vertices does not admit a drawing with  $\vartheta(C_4) = 2$  crossings. As reported in [51], Conway conjectured in 1969 that each thrackable graph contains at most as many edges as vertices. His conjecture, known as *Conway's thrackle conjecture*, still remains open. Woodall [51] was the first to work on the thrackle conjecture and, assuming the conjecture was true, showed that a finite graph admits a thrackle if and only if it contains at most one odd cycle, no 4-cycle, and each of its connected components is either a tree or it contains exactly one cycle. Refer to [26, 27, 41, 52] and the references therein for progress relevant to Conway's Thrackle conjecture. Graph classes that admit thrackles (and satisfy the thrackle conjecture) include cycles of more than 4 vertices [36] and trees [48]. Given a tree  $T$  and an integer  $0 \leq \chi \leq \vartheta(T)$ , Piazza et al. [48] presented an algorithm that constructs a drawing of  $T$  with  $\chi$  crossings and curve complexity  $O(n)$ . A similar result was presented by Harborth [36] for any cycle  $C_n$ ,  $n > 4$ . Hence, the algorithms in [48] and [36] compute drawings that satisfy Property 3.

Relationship between Properties 2 and 3 – A *linear thrackle* is a thrackle with curve complexity zero. Not all graphs admit linear thrackles. For example, a cycle  $C_6$  with 6 edges does not admit a linear thrackle with  $\vartheta(C_6) = 9$  crossings (but it admits a thrackle [48]). It should be noted that, as proved by Erdős [24] and Perles [47], Conway's conjecture holds for linear thrackles. More generally, there is a complex relationship between number of

crossings and curve complexity. For example, a cycle  $C$  with an odd number of vertices admits a drawing with curve complexity zero and  $\chi$  crossings, where  $\chi$  is any number up to  $\vartheta(C)$  except  $\vartheta(C) - 1$  [28]. On the other hand, an  $n$ -vertex cycle  $C$  with an even number of vertices admits a drawing with complexity zero and  $\chi$  crossings, where  $\chi$  is any number up to  $n(n-4)/2 + 1 = \vartheta(C) - n/2 + 1$  [28]. Notice that there exist planar graphs admitting straight-line pointset embeddings on non-convex pointsets that have more crossings than any straight-line pointset embedding on a pointset in convex position [14]

Besides the above properties we are also interested in producing RAC drawings. A seminal user-study by Huang, Eades, and Hong [37] shows that edge crossings that form sharp angles affect the readability of a drawing much more than those that form large angles. This has motivated the study of right-angle crossing drawings (*RAC drawings*), that are graph drawings where the crossing angles are all  $\frac{\pi}{2}$ ; see, e.g., [1, 2, 6, 16, 18, 19, 17, 20, 21, 22, 23] for surveys and papers about RAC drawings. Point-set RAC embeddings were studied by Fink et al. [25] who, among other results, show that every graph admits a point-set RAC embedding with curve complexity 3.

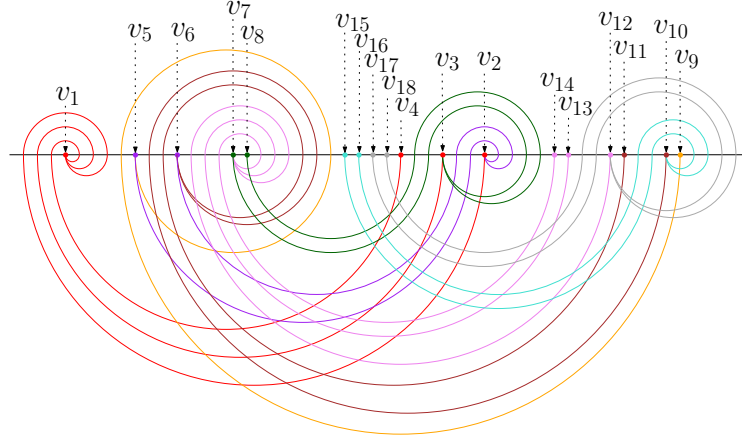
**Our Contribution.** We solve the main problem in this paper for any  $0 \leq \chi \leq \vartheta(T)$ . To this aim, we first present an  $O(n)$ -time algorithm which *tangles*  $T$  by computing a linear layout with  $\vartheta(T)$  edge crossings where each edge traverses the spine (i.e. the line passing through the vertices) a constant number of times. We then show how to *untangle* the linear layout until we reach the desired number  $\chi$  of edge crossings. By carefully *pruning* some edges of  $T$ , such a linear layout can be computed in  $O(n^2)$  time. If  $T$  is a path the time to construct the linear layout reduces to  $O(n)$ . Finally, the linear layout is used to construct a point-set embedding  $\Gamma$  of  $T$  having  $\chi$  edge crossings and curve complexity at most 5. If we require  $\Gamma$  to be a RAC drawing, then  $\Gamma$  has curve complexity at most 9, which decreases to 6 if the points are collinear. For the case of paths, a RAC point-set embedding with curve complexity 3 can be computed in  $O(n \log n)$  time.

Full proofs of statements marked with a “ $\star$ ” can be found in [5].

## 2 Preliminaries

We only consider *simple* graphs, i.e. graphs not containing loops and multiple edges. A *drawing* of a graph  $G(V, E)$  maps each vertex  $v \in V$  to a distinct point on the plane and each edge  $(u, v) \in E$  to a Jordan arc joining the points representing  $u$  and  $v$ . A drawing is *simple* if: (i) two adjacent edges do not intersect except at a common end-vertex; (ii) no three edges intersect on the same point; (iii) two edges intersect at most once at a common interior point; and (iv) the intersection between a pair of edges must be “transverse”, i.e., the curves of the two edges must alternate around the intersection point. We only consider simple drawings of graphs which we shall call just *drawings*, for short. A drawing is *straight-line* if the edges are represented as straight-line segments. A drawing is *polyline* if each edge is represented as a chain of straight-line segments such that any consecutive segments of the chain have different slopes. The *curve complexity* of a polyline drawing is the maximum number of bends per edge in the drawing. A drawing is a *RAC drawing* if any two edges that cross do so by forming four  $\frac{\pi}{2}$  angles around the crossing point.

A *topological linear embedding*  $\mathcal{E}$  of graph  $G(V, E)$  is defined as follows. (1) Each edge  $(u, v)$  of  $E$  is mapped to a *subdivision path*  $\pi$  composed of  $k > 0$  *legs*  $\langle (u_0, u_1), (u_1, u_2), \dots, (u_{k-1}, u_k) \rangle$  with  $u_0 = u$  and  $u_k = v$ , possibly consisting of a single leg  $(u, v)$ . The *subdivision vertices*  $u_1, u_2, \dots, u_{k-1}$  internal to  $\pi$  are *spine traversals*. (2) The vertices in  $V$  and the



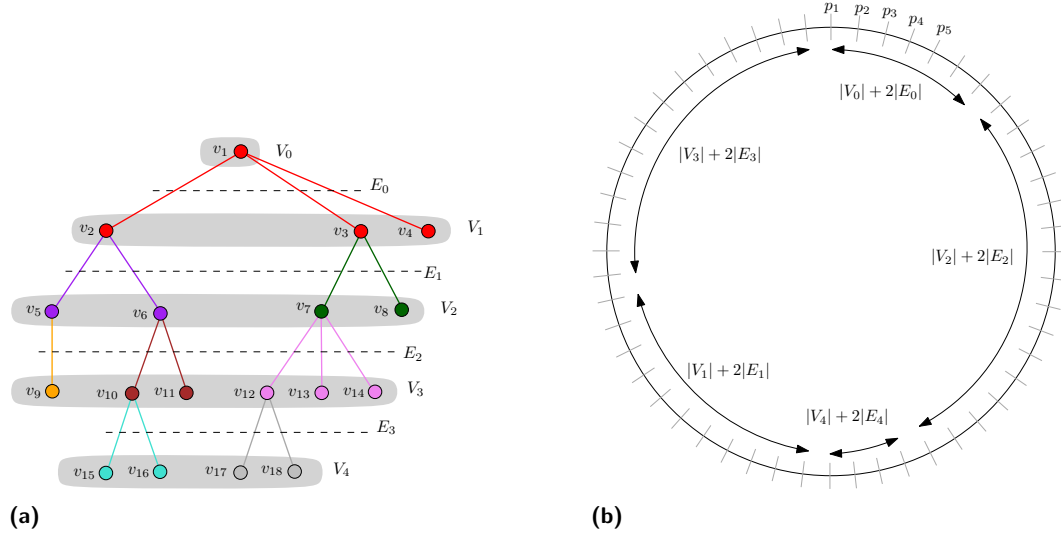
■ **Figure 1** A topological linear layout of the tree of Fig. 2a obtained from the topological circular layout of Fig. 4.

spine traversals are ordered in  $\mathcal{E}$ ; this order is the *spine* of  $\mathcal{E}$ . (3) The legs are subdivided into two partition sets named *top page* and *bottom page* in such a way that any two consecutive legs of a subdivision path are assigned to distinct partition sets. (4) Two legs  $(a, b)$  and  $(c, d)$  in the same partition set *cross* if  $a \succ c \succ b \succ d$ . (4.1) No two legs of the same edge can cross. (4.2) Edges  $(u, v)$  and  $(w, z)$  cross  $\rho$  times if there are  $\rho$  distinct pairs of their legs that cross. We impose  $\rho \leq 1$ . If  $\rho = 0$   $(u, v)$  and  $(w, z)$  *do not cross* in  $\mathcal{E}$ , whereas, if  $\rho = 1$  they do.

A drawing  $\Gamma$  of a topological linear embedding  $\mathcal{E}$  can be constructed as follows. The vertices and the spine traversals of  $\mathcal{E}$  are represented in  $\Gamma$  as points along a horizontal line, called *spine* of  $\Gamma$ , in the same left-to-right order they have in  $\mathcal{E}$ . The half-plane above (resp. below) the spine is the *top page* (resp. *bottom page*) of  $\Gamma$ . A leg  $(a, b)$  of  $G$  assigned to the top page (resp. bottom page) in  $\mathcal{E}$  is represented as the arc of a circumference in the top page (resp. bottom page) of  $\Gamma$  having the points corresponding to  $a$  and  $b$  along the spine as the extreme points of the diameter. Such arc is a *leg* of  $\Gamma$ . As a consequence, two legs cross in  $\Gamma$  if and only if they cross in  $\mathcal{E}$ . If  $a$  is a spine traversal in the topological linear embedding, the point representing  $a$  along the spine of  $\Gamma$  is a *spine traversal* of  $\Gamma$ . Drawing  $\Gamma$  is a *topological linear layout* of  $G$ . See Fig. 1. It is immediate to see that, given a topological linear embedding of a graph  $G$ , a topological linear layout of  $G$  can be constructed in  $O(n + \sigma m)$  time, where  $n$ ,  $m$  and  $\sigma$  denote the number of vertices, edges, and maximum number of spine traversals per edge of  $G$ , respectively.

### 3 Topological Linear Embeddings of Trees

In this section we describe Algorithm TANGLE, that computes in  $O(n)$  time a topological linear embedding of an  $n$ -vertex tree  $T$  with  $\vartheta(T)$  crossings and with two subdivision vertices per edge. Second, we describe Algorithm UNTANGLE which can be iteratively used to reduce the number of crossings from  $\vartheta(T)$  down to zero, hence guaranteeing the existence of a topological linear embedding with  $\chi$  crossings,  $\chi \in [1.. \vartheta(T)]$  and with at most two subdivisions per edge. Since  $\vartheta(T) \in O(n^2)$  and each step of UNTANGLE takes  $O(n)$  time, this leads to an  $O(n^3)$ -time algorithm to produce a topological linear embedding with  $\chi$  crossings,  $\chi \in [1.. \vartheta(T)]$ . We will also show how to reduce this time to  $O(n^2)$ .



■ **Figure 2** An example tree (a) and the subsequences of points on  $\mathcal{C}$  used for its vertices/edges (b).

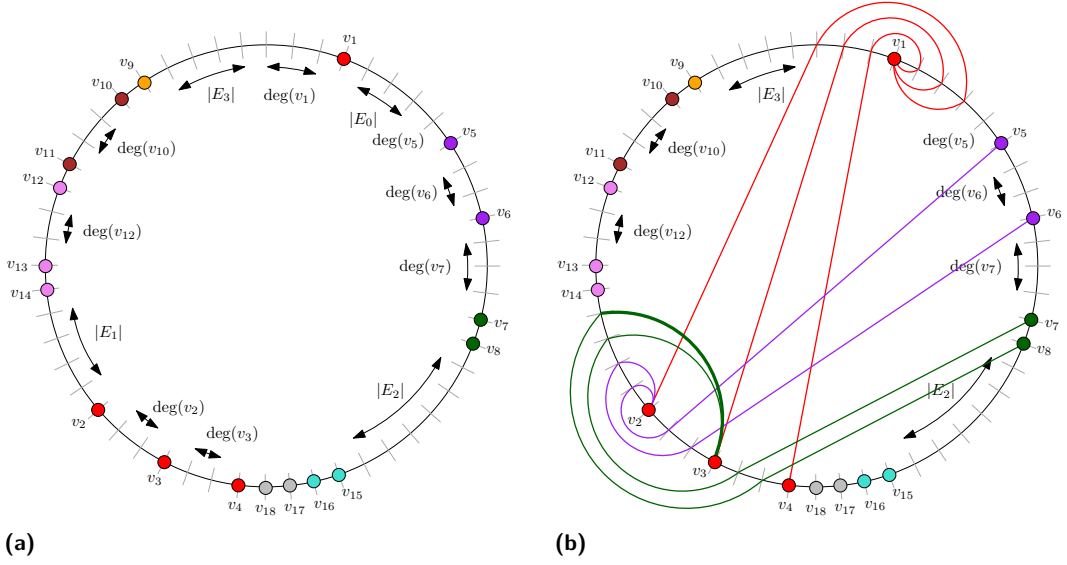
For the ease of the description, instead of producing a topological linear embedding, we produce a *topological circular embedding* whose definition is analogous to that of a topological linear embedding with the difference that the spine is a circular ordering<sup>1</sup> rather than a linear ordering. A topological linear embedding is obtained from a topological circular embedding and vice versa by considering the spine ordering as linear or circular, respectively. Observe that crossings among the edges and spine traversals are preserved in the two embeddings.

### 3.1 Algorithm Tangle

In the topological circular embedding  $\mathcal{E}_o$  of tree  $T(V, E)$  constructed by Algorithm TANGLE each edge  $e \in E$  has two subdivision vertices and, hence, it is split into three legs denoted by  $\overline{e}^\circ$ ,  $\overline{e}$ , and  $\overline{e}^\circ$ , where  $\overline{e}^\circ$  is incident to the parent vertex and  $\overline{e}^\circ$  is incident to the child vertex. We will assign  $\overline{e}^\circ$  and  $\overline{e}^\circ$  to the inner region (bottom page) and second leg  $\overline{e}$  to the outer region (top page). The crossing among legs will happen exclusively in the inner region.

We assume that  $T$  is rooted at vertex  $v_1$  and planarly embedded in such a way that for any vertex  $v_i$ , with  $i = 1, \dots, n$ , the non-leaf children of  $v_i$  precede the leaf children of  $v_i$  (see, for example, Fig. 2a). Let  $h$  be the height of  $T$ . For  $j = 0, \dots, h$ , we denote by  $V_j$  the set of the vertices at depth  $j$  in  $T$  and by  $E_j$ ,  $j = 0, \dots, h-1$ , the edges between the vertices in  $V_j$  and those in  $V_{j+1}$ . We choose a sequence  $P = \langle p_1, p_2, \dots, p_{|V|+2|E|} \rangle$  of  $|V| + 2|E|$  distinct points in this clockwise order along a cycle  $\mathcal{C}$ . The points in  $P$  will be used either by the vertices in  $V$  or by the subdivision vertices of the edges in  $E$ . We subdivide the points in  $P$  into contiguous subsequences of points along  $\mathcal{C}$ . More specifically, if  $h$  is even, we subdivide  $P$  into  $P_0, P_2, P_4, \dots, P_h, P_1, P_3, P_5, \dots, P_{h-1}$ ; otherwise, we subdivide  $P$  into  $P_0, P_2, P_4, \dots, P_{h-1}, P_1, P_3, P_5, \dots, P_h$ . Each subsequence  $P_j$  ( $j = 0, 1, \dots, h$ ) has size  $|P_j| = |V_j| + 2|E_j|$  and is used by the vertices in  $V_j$  and by the subdivision vertices of the edges in  $E_j$  (refer to Fig. 2b).

<sup>1</sup> A circular ordering generalizes a linear ordering to settings where there is no “first” or “last” element; we can think it as arranging elements around a circle rather than along a line.



■ **Figure 3** Placement of the vertices (a) and Edge routing (b) of Algorithm TANGLE.

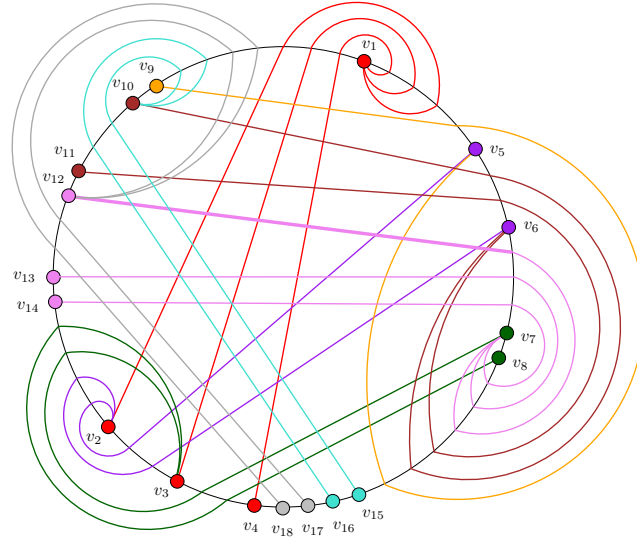
**Vertex placement.** We map the vertices of  $V_j$  to specific points of  $P_j$ . If  $j$  is even (resp.  $j$  is odd), denote by  $v_{j,1}, v_{j,2}, \dots, v_{j,|V_j|}$  the vertices of  $V_j$  in the order (resp. in the reversed order) as they appear at depth  $j$  in  $T$ . For  $k = 1, \dots, |V_j|$ , we leave  $\text{children}(v_{j,k})$  points of  $P_j$  unused and then we place  $v_{j,k}$  (see Fig. 3a). Observe that after the last vertex  $v_{j,|V_j|}$  of  $V_j$  has been placed, there are  $2|E_j|$  unused points,  $|E_j|$  of which are at the end of  $P_j$ .

**Edge routing.** We describe how to embed the edges of  $E_j$ , for each  $j = 0, \dots, h-1$ . Denote by  $p_{j,1}, p_{j,2}, \dots, p_{j,|V_j|+2|E_j|}$  the points of  $P_j$  in this clockwise order around  $\mathcal{C}$ . Recall that the vertices of  $V_j$  have been already placed in  $P_j$  and that each vertex  $v_{j,k}$ , with  $k = 1, \dots, |V_j|$ , is preceded by  $\text{children}(v_{j,k})$  unused points, while the whole sequence  $P_j$  is closed by  $|E_j|$  unused points. For  $k = 1, \dots, |V_j|$ , we process vertex  $v_{j,k}$  of  $V_j$  and, for  $\ell = 1, \dots, \text{children}(v_{j,k})$ , we embed each edge  $e_\ell \in |E_j|$  incident to  $v_{j,k}$ , where edges in  $E_j$  are considered in the left to right order as they appear in the embedding of  $T$ . Intuitively, edge  $e_\ell$  will greedily use the last unused point of  $P_j$  and the first unused point of  $P_j$ . More formally, we embed  $\overrightarrow{e_\ell}$  in the inner region using as its endpoint the last unused point of  $P_j$  (refer to Fig. 3b); we embed  $\overleftarrow{e_\ell}$  in the outer region using as its second endpoint the  $(\text{children}(v_{j,k}) - \ell + 1)$ -th point of  $P_j$  clockwise preceding the point used by  $v_{j,k}$  (which is also the first unused point of  $P_j$ ); we embed  $\overleftrightarrow{e_\ell}$  in the inner region hitting the point of  $P_{j+1}$  where the  $\ell$ -th child of  $v_{j,k}$  lies.

► **Lemma 1.** *Algorithm TANGLE computes in  $O(n)$  time a topological circular embedding of an  $n$ -vertex tree  $T$  with  $\vartheta(T)$  crossings where every edge traverses the spine exactly twice.*

**Proof sketch.** By construction, all the edges of a topological circular embedding  $\mathcal{E}_o$  produced by Algorithm TANGLE are composed by three legs, the second of which is drawn in the outer region. It can be easily shown that the second legs of the edges in  $E$  do not cross in  $\mathcal{E}_o$ . Also, the first legs of the edges in  $E$  do not cross among themselves in  $\mathcal{E}_o$ . It follows that the only possible crossings in  $\mathcal{E}_o$  are between a first leg and a third leg of two edges in  $E$  or between two third legs of two edges in  $E$ . Consider two edges  $(u, v)$  and  $(w, z)$  of the same  $E_j$ ,  $j \in [0..h-1]$ , where  $u$  and  $w$  belong to  $P_j$ . Edges  $(u, v)$  and  $(w, z)$  cross whenever  $u$  uses a point of  $P_j$  that precedes the point of  $P_j$  used by  $w$  ( $\overrightarrow{(u, v)}$  crosses  $\overrightarrow{(w, z)}$ ). Hence, all the





■ **Figure 4** A drawing representing a topological circular embedding of the tree  $T$  of Fig. 2a with  $\vartheta(T)$  edge crossings and two spine crossings per edge.

edges of  $E_j$  cross among themselves with the exception of those that are incident to the same vertex of  $V_j$ . Consider two edges  $(u, v) \in E_j$  and  $(w, z) \in E_{j-1}$ , with  $j \in [1..h-1]$ , where  $u$  and  $z$  belong to  $V_j$ . Edges  $(u, v)$  and  $(w, z)$  cross when  $u$  uses a point of  $P_j$  that precedes the point of  $P_j$  used by  $z$  ( $(u, v)$  crosses  $(w, z)$ ). Also,  $(u, v)$  and  $(w, z)$  cross when  $u$  uses a point of  $P_j$  that follows the point of  $P_j$  used by  $z$  ( $(u, v)$  crosses  $(w, z)$ ). Hence,  $(u, v) \in E_j$  and  $(w, z) \in E_{j-1}$  cross whenever they are not adjacent (i.e., whenever  $u \neq z$ ). Finally, two edges  $(u, v) \in E_j$  and  $(w, z) \in E_{j'}$  with  $j' \notin \{j-1, j, j+1\}$  always cross once ( $(u, v)$  crosses  $(w, z)$ ). In conclusion two edges of  $T$  cross if and only if they are not adjacent. Hence the number of crossings is  $\vartheta(T)$ . It is easy to check that  $\mathcal{E}_o$  can be computed in  $O(n)$  time. ◀

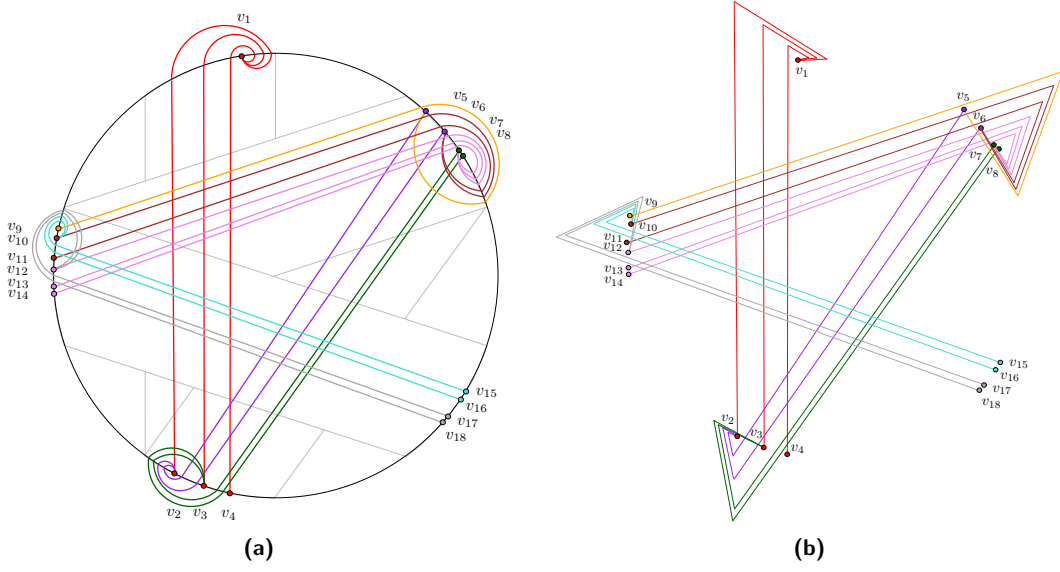
An example drawing of the topological circular embedding computed by Algorithm TANGLE is shown in Fig. 4. As a byproduct of the results in this section, we point out that from the circular topological embedding  $\mathcal{E}_o$  it is not difficult to compute a polyline drawing of  $T$  with  $\vartheta(T)$  crossings (a thrackle of  $T$ ) with curve complexity two as in Fig. 5.

### 3.2 Algorithm Untangle

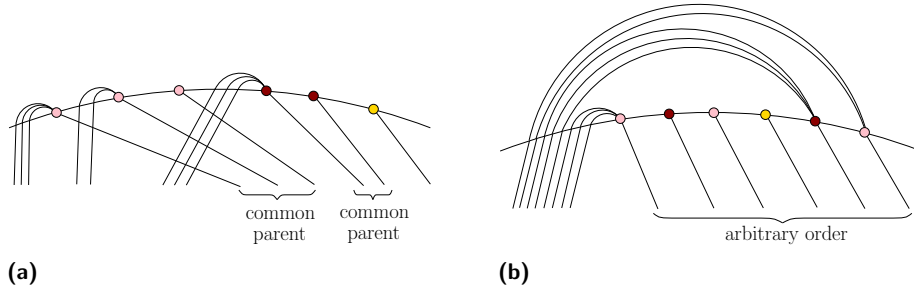
We describe Algorithm UNTANGLE as if the target were to reduce the number of crossings down to zero. In order to obtain a topological circular embedding  $\mathcal{E}_o^\chi$  with  $\chi$  crossings it is sufficient to run  $\vartheta(T) - \chi$  iterations of the algorithm.

Let  $\vartheta'(T) = \frac{1}{2} \cdot |E|^2 - \frac{1}{2} \cdot \sum_{i=0}^{h-1} |E_i|^2 - \sum_{i=0}^{h-2} (|E_i| \cdot |E_{i+1}|)$ . The algorithm has two phases: in the first phase the crossings are reduced from  $\vartheta(T)$  to  $\vartheta'(T)$ , obtaining a topological circular embedding  $\mathcal{E}_o^{\vartheta'(T)}$  where each edge is subdivided exactly once. The reason for the choice of  $\vartheta'(T)$  will become evident soon. In the second phase the number of crossings is reduced to zero.

**Phase 1: reducing the crossings to  $\vartheta'(T)$ .** Let  $\mathcal{E}_o^{\vartheta(T)}$  be a topological circular embedding of  $T$  with  $\vartheta(T)$  crossings. While reducing at each step the number of crossings by one, we process  $T$  level by level bottom-up. In particular, for each  $i = h-1, h-2, \dots, 0$ , we first



■ **Figure 5** From a topological circular embedding of a tree to a thrackle with curve complexity two.



■ **Figure 6** Examples of vertices embedded as rainbows (a) and a full-rainbow configuration (b).

modify the embedding of  $T$  in such a way that each vertex of  $V_i$  assumes a configuration that we call “rainbow” and we then bring all vertices of  $V_i$  and edges of  $E_i$  to a configuration that we call “full-rainbow”. When all levels of the tree are processed, the number of crossings will be exactly  $\vartheta'(T)$  and each edge of  $T$  will have exactly one subdivision vertex.

A vertex  $v$  is embedded as a *rainbow* if: (i) the edges that lead to its children are subdivided exactly once, where their first leg is assigned to the outer region and their second leg is assigned to the inner region and (ii) the subdivision vertex of such edges immediately precede  $v$  in clockwise order. See Fig. 6a for an example of vertices embedded as rainbows. The vertices of  $V_i$  are in a *full-rainbow configuration* if: (i) the edges of  $E_i$  are planar drawn with only two legs, the first external and the second internal to  $\mathcal{C}$  and (ii) the crossing of an edge  $(u, v)$  of  $E_i$  with  $\mathcal{C}$  clockwise precedes all the vertices of  $V_i$ . See Fig. 6b for an example of a full-rainbow configuration. Observe that in a full-rainbow configuration of  $V_i$  the first legs of edges of  $E_i$  do not cross each other and no edge of  $E_i$  crosses an edge of  $E_{i-1}$ .

There are as many full-rainbow configurations of  $V_i$  as there are permutations of the vertices in  $V_i$ . In particular, if we have a full-rainbow configuration and we swap the position of two consecutive vertices  $v_x$  and  $v_y$ , we can obtain another full-rainbow configuration by swapping the position of the crossings with  $\mathcal{C}$  of the edges of  $E_i$  incident to  $v_x$  with those of  $E_i$  incident to  $v_y$ , and propagating the swap to the descendants of  $v_x$  and  $v_y$ . Hence, we obtain the following lemma:

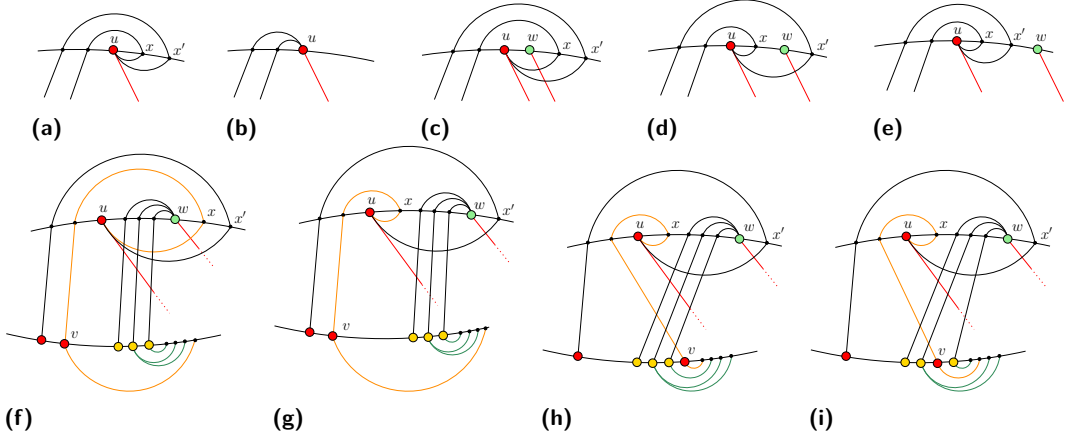


► **Lemma 2.** *Let  $T$  be a tree of height  $h$ . Let  $0 \leq i \leq h$  and assume that for all  $j \geq i$  the vertices of  $V_j$  are in a full-rainbow configuration. There exists a topological circular embedding  $\mathcal{E}_o$  of  $T$  such that the only crossings among edges in  $E_i \cup E_{i+1} \cup \dots \cup E_{h-1}$  occur between edges in  $E_j$  and  $E_k$  with  $k > j + 1$ .*

Note that for  $i = 0$  Lemma 2 yields a topological circular embedding  $\mathcal{E}_o$  of  $T$  where the edges of the same  $E_i$  do not cross and edges of  $E_i$  and  $E_{i+1}$  do not cross either. As indicated in Lemma 3, such an embedding has exactly  $\vartheta'(T)$  crossings.

Suppose to have a topological circular embedding  $\mathcal{E}_o^x$  of  $T$  such that for a given  $0 \leq i \leq h-1$  it holds that: (i) for all  $j > i$  the vertices of  $V_j$  are in a full-rainbow configuration; (ii) for all  $j \leq i$  the vertices of  $V_j$  are embedded as in  $\mathcal{E}_o^{\vartheta(T)}$ ; and (iii) for all  $j < i$  the edges of  $E_j$  are embedded as in  $\mathcal{E}_o^{\vartheta(T)}$ . Observe that for  $i = h-1$  such a drawing is trivially  $\mathcal{E}_o^{\vartheta(T)}$ . First, we reduce the number of crossings one by one moving all vertices of  $V_i$  to a rainbow configuration. Second, we reduce the crossings one by one moving  $V_i$  and  $E_i$  to a full-rainbow configuration. For the first target, recall that in  $\mathcal{E}_o^{\vartheta(T)}$  a second leg of an edge in  $E_i$  is embedded on the outer region and that the second legs of edges in  $E_i$  do not cross. Consider a non-leaf vertex  $u$  of  $V_i$  and one of its edges  $e = (u, v) \in E_i$ . If  $u$  has more than one child in  $T$ , then consider the edge  $e \in E_i$  such that the first subdivision vertex  $x$  of  $e$  is closer to  $u$  (refer to Fig. 7). There are three cases: Case 1: If the last point of  $P_i$  before  $x$  in clockwise order is used by  $u$  (see Fig. 7a), then the first subdivision vertex of the edges joining  $u$  with its children can be removed, and  $u$  can be joined directly to their second subdivision vertices on the outer region without changing the total number of crossings (see Fig. 7b). This yields an embedding of  $u$  as a rainbow. Case 2: If the last point of  $P_i$  before  $x$  in clockwise order is used by a leaf  $w$  (see Fig. 7c), then we can swap the position of  $w$  and  $x$ , decreasing the number of crossings by one, having removed the crossing that was between the edge incident to  $w$  and  $e$  (see Fig. 7d). This swap operation can be repeated with the other subdivision vertices  $x', x'', \dots$  of the edges joining  $u$  with its children, reducing each time the number of crossings by one (see, for example, Fig. 7e). Case 3: If the last point of  $P_i$  before  $x$  in clockwise order is used by a non-leaf vertex  $w$ , then we can assume that  $w$  is embedded as a rainbow (see Fig. 7f). Indeed, if this was not the case we would process  $w$  before  $u$ . We move  $w$ , together with the first subdivision vertices of the edges joining  $w$  with its children, clockwise after  $x$  (see Fig. 7g). In doing so, we reduce the number of crossings by  $\deg(w)$  instead of one unit (compare Figs. 7f and 7g where  $(u, v)$  loses four crossings). However, since the vertices of  $V_{i+1}$  are in a full-rainbow configuration, they can be permuted and, in particular, the vertex  $v'$  that clockwise follows the children of  $w$  can be moved counter-clockwise so that the third leg of edge  $(u, v')$  crosses children( $w$ ) edges of  $E_i$  and the total number of crossings is reduced by one unit only (see Fig. 7h where vertex  $v'$  coincides with  $v$ ). Successively, we move  $v'$  back to its original position reducing the crossings of one unit at each swap (Fig. 7i shows the first step of such a process). We iterate these operations until none of the above moves apply, i.e., until all the vertices of  $V_i$  are in a rainbow configuration.

We now describe how to change the embedding of  $T$  in such a way that the vertices in  $V_i$  and the edges in  $E_i$  are in a full-rainbow configuration. Recall that, since each vertex of  $V_i$  has a rainbow configuration, each edge of  $E_i$  has a single subdivision vertex. Consider the first subdivision vertex  $x$ , in clockwise order, of an edge  $e \in E_i$  that is preceded by a vertex  $u \in V_i$ . Move  $x$  before  $u$  and before all subdivision vertices of the edges joining  $u$  with its children. Observe that this change reduces the number of crossings by one (the crossing between  $e$  and the edge of  $E_{i-1}$  incident to  $u$ ). By iterating this process we have that no subdivision vertex of  $E_i$  is preceded by a vertex of  $V_i$  in  $P_i$ , i.e., the vertices of  $V_i$  are in a full-rainbow configuration.



■ **Figure 7** Illustrations for Algorithm UNTANGLE.

► **Lemma 3.** *Following the completion of Phase 1, when all  $V_i$ ,  $0 \leq i \leq h$ , are in full-rainbow configuration, the total number of crossings is  $\vartheta'(T)$ .*

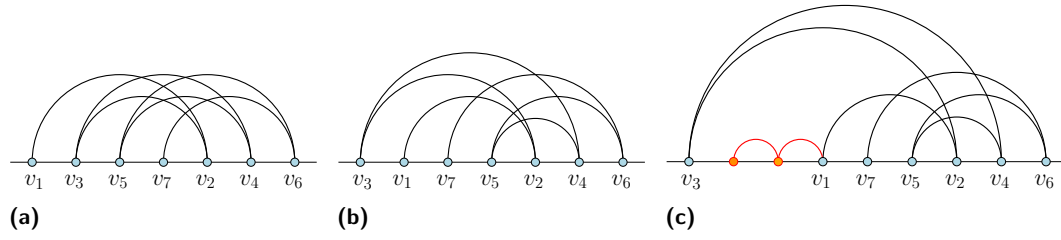
**Phase 2: reducing the crossings to zero.** When all the levels of the tree are in a full-rainbow configuration, all edges are drawn with a single subdivision vertex. Consider an edge  $(u, v)$  incident to a vertex  $v \in V_h$  such that  $v$  is the last vertex clockwise of  $V_h$ . Observe that  $v$  is a leaf. Denote by  $x$  the subdivision vertex of  $(u, v)$ . Vertex  $v$  can be moved clockwise towards  $x$  removing one crossing at each step. When  $v$  reaches  $x$ , vertex  $v$  can be transferred close to its parent  $u$  and identified with it (i.e.,  $v$  can be planarly embedded arbitrarily close to  $u$  and moved with  $u$  from now on). Iteratively performing this operation on the last vertex of the last level of  $T$  reduces the number of crossings down to zero, thus we have:

► **Lemma 4.** *Any  $n$ -vertex tree  $T$  admits a topological linear embedding with  $\chi$  edge crossings,  $\chi \in [0, \vartheta(T)]$ , where every edge traverses the spine at most twice. Such an embedding can be computed in  $O(n^3)$  time.*

### 3.3 Algorithm Prune-Tangle-Untangle

In this section we describe an algorithm, called PRUNE-TANGLE-UNTANGLE, that reduces the time complexity of Lemma 4 to  $O(n^2)$ . The algorithm uses the TANGLE and UNTANGLE algorithms. However, before these, it launches the PRUNE procedure illustrated below.

Let  $T_n$  be the input  $n$ -vertex tree. The leaves of  $T_n$  are recursively removed (in any order) obtaining trees  $T_{n-1}, T_{n-2}, \dots$ , while  $\vartheta(T_{n-i}) > \chi$ . Note that an edge  $(u, v)$  of an  $n$ -vertex tree  $T_n$ , incident to a leaf  $v$ , may cross all other  $n - 2$  edges of  $T_n$  except the  $\deg(u) - 1$  edges incident to  $u$ . Hence, removing  $(u, v)$  produces a tree  $T_{n-1}$  where the number of possible crossings decreases by a quantity in  $[0..n]$ . To efficiently identify the leaves and compute the current number of crossings, we equip vertices with their degrees and maintain a list of all the leaves. When a leaf is removed from the tree (and from the list) the degree of its parent is decreased, and if it becomes one, the parent is added to the list of leaves. We continue until we encounter a leaf that cannot be removed. Thus, it holds that  $\vartheta(T_{n-i}) < \chi + n$ . Then, we launch Algorithm TANGLE on  $T_{n-i}$  to obtain  $\mathcal{E}_o^{\vartheta(T_{n-i})}$  of  $T_{n-i}$  and then perform  $\vartheta(T_{n-i}) - \chi$  iterations of UNTANGLE to obtain  $\mathcal{E}_o^\chi$  of  $T_{n-i}$ . Finally, the removed edges are planarly added in reverse order as they were removed, iteratively placing them along  $\mathcal{C}$  at a small enough distance from their parent, to obtain a topological circular embedding of  $T$  with  $\chi$  crossings. The above description implies the following:



**Figure 8** Topological linear layouts of paths. (a) A linear layout of  $P_7$  with 10 crossings. (b) Linear layout obtained moving vertices  $v_1$  and  $v_7$  in the one in Fig. 8a.  $\Delta_\chi = 2$  and  $k = 1$ . (c) Linear layout computed for path  $P_9$  and  $\chi = 8$ .

► **Theorem 5.** Any  $n$ -vertex tree  $T$  admits a topological linear embedding with  $\chi$  edge crossings,  $\chi \in [0..v(T)]$ , where every edge traverses the spine at most twice. Such an embedding can be computed in  $O(n^2)$  time.

#### 4 Topological Linear Embeddings of Paths

In the first part of this section we present an algorithm, which we call  $\chi$ -Spiral that, given an  $n$ -vertex ( $n \geq 4$ ) path  $P_n$  and any integer  $\chi$  such that  $(n-3)(n-4)/2 < \chi \leq (n-2)(n-3)/2$  computes a spine traversal free, one-page topological linear embedding  $\Gamma$  with  $\chi$  crossings of  $P_n$ . In all topological linear embedding constructed in this section each edge is mapped to just one leg. Hence, we refer to edges and not to legs. Notice also that  $(n-3)(n-4)/2 = v(P_{n-1})$  is exactly the maximum number of crossings for a drawing of a path with  $n-1$  vertices.

Algorithm  $\chi$ -Spiral consists of two steps.

**(Step 1)** Compute a spine traversal free, one-page topological linear embedding  $\Gamma$  with  $\chi' = v(P_n) = (n-2)(n-3)/2$  crossings of  $P_n$ . Let  $v_1, \dots, v_n$  be the vertices of  $P_n$  each with a subscript that corresponds to the order in which it appears by visiting  $P_n$  from any of its vertices of degree one to the other. Place on the spine of  $\Gamma$  first the vertices with odd subscript, in increasing order of subscript, and then the vertices with even subscript, in increasing order of subscript, too. Hence, the order of the vertices on the spine of  $\Gamma$  is  $v_1 \prec v_3 \prec \dots \prec v_{n-1} \prec v_2 \prec v_4 \prec \dots \prec v_n$  if  $n$  is even and  $v_1 \prec v_3 \prec \dots \prec v_n \prec v_2 \prec v_4 \prec \dots \prec v_{n-1}$  if  $n$  is odd. Then, assign all the edges of  $P_n$  to the same page, say top page, of  $\Gamma$ . The linear embedding computed by Step 1 of Algorithm  $\chi$ -Spiral for path  $P_7$  is shown in Fig. 8a

**(Step 2)** Let  $\Delta_\chi = \chi' - \chi$  be the “excess of crossings” that  $\Gamma$  has with respect to the target  $\chi$ . Observe that  $\Delta_\chi < n-3$ . Now, perform what follows. If  $\Delta_\chi = 2k+1$  is odd, then move vertex  $v_1$  by  $k+1$  positions on the spine to the right, leaving the position of all the other vertices unchanged. Else, ( $\Delta_\chi = 2k$  is even) move vertex  $v_1$  on the spine by  $k$  positions to the right, leaving the position of all the other vertices unchanged. Then, as a last action of the even case, move vertex  $v_n$  to the left by one position. The assignment of all the edges to the same page remains unchanged. The topological linear embedding computed by Algorithm  $\chi$ -Spiral for path  $P_7$  and  $\chi = 8$  (i.e.,  $\Delta_\chi = 2$ ) is shown in Fig. 8b.

► **Lemma 6.** Let  $P_n$  be an  $n$ -vertex ( $n \geq 4$ ) path and let  $\chi$  be an integer such that  $(n-3)(n-4)/2 < \chi \leq (n-2)(n-3)/2$ . Algorithm  $\chi$ -Spiral computes a spine traversal free, one-page topological linear embedding  $\Gamma$  with  $\chi$  crossings of  $P_n$  in  $O(n)$  time.

**Proof sketch.** First, we prove that at the end of Step 1 Algorithm  $\chi$ -Spiral gets a topological linear embedding  $\Gamma$  with  $\chi' = (n-2)(n-3)/2$  crossings. Since all the edges of  $P_n$  are assigned to the same page,  $\Gamma$  is one-page and spine traversal free. We prove that, in  $\Gamma$ , each edge crosses all the other edges but those incident to its end-vertices. Consider any two edges  $(v_i, v_{i+1})$  and  $(v_j, v_{j+1})$  of  $P_m$  such that  $i, j, i+1$ , and  $j+1$  are pairwise different and assume, w.l.o.g., that  $i < j$ . We prove that  $(v_i, v_{i+1})$  and  $(v_j, v_{j+1})$  cross in  $\Gamma$ . Suppose that  $i$  and  $j$  are both odd, the other cases being similar. We have that, since  $i < j$ ,  $v_i \prec v_j$ . Also, both  $i+1$  and  $j+1$  are even, with  $i+1 < j+1$ , hence,  $v_{i+1} \prec v_{j+1}$ . Further, since  $j$  is odd and  $i+1$  is even,  $v_j$  is placed before  $v_{i+1}$  on the spine ( $v_j \prec v_{i+1}$ ). In summary,  $v_i \prec v_j \prec v_{i+1} \prec v_{j+1}$ . It follows that  $(v_i, v_{i+1})$  and  $(v_j, v_{j+1})$  cross. The time complexity of Step 1 of the algorithm follows from the construction, which is just a scan of the path.

Concerning Step 2 of the algorithm, we have what follows. (1) Because of the range of  $\chi$ , we have that  $\Delta_\chi < n-3$ . (2) If  $\Delta_\chi$  is odd we have  $0 \leq k < (n-4)/2$ . If  $\Delta_\chi$  is even we have  $0 \leq k < (n-3)/2$ . Because of these upper bounds on  $k$ , after the moves,  $v_1$  is still to the left of  $v_n$  in  $\Gamma$ . (3) If  $v_1$  moves by one position to the right, then it is placed right after  $v_3$ , hence, the number of crossings in  $\Gamma$  decreases by one (namely  $(v_1, v_2)$  no longer crosses  $(v_3, v_4)$ ). (4) If  $v_1$  moves by  $k+1$  positions to the right then the number of crossings in  $\Gamma$  decreases by  $1+2k$ ; e.g. if  $k=1$  then  $v_1$  is positioned between  $v_5$  and  $v_7$ , hence,  $(v_1, v_2)$  no longer crosses  $(v_3, v_4)$ ,  $(v_4, v_5)$ , and  $(v_5, v_6)$ . Essentially, the first move of  $v_1$  to the right decreases the number of crossings in  $\Gamma$  by one, while each of the subsequent moves decreases the number of crossings by exactly two. Observe also that a move of  $v_n$  to the left of one position decreases the number of crossings in  $\Gamma$  by one unit. Since Step 2 involves only a linear number of moves on the spine of  $\Gamma$ , its time complexity is also linear. ◀

Now, we exploit Algorithm  $\chi$ -spiral and Lemma 6 to efficiently produce a drawing of  $P_n$  with “any” number of crossings, proving the following lemma.

► **Lemma 7.** *Let  $P_n$  be an  $n$ -vertex path. For each  $\chi \in [0, \vartheta(P_n)]$ ,  $P_n$  admits a one-page topological linear embedding with  $\chi$  crossings, that can be computed in  $O(n)$  time.*

**Proof.** In order to compute the desired linear embedding, we first look for an integer  $n'$  such that  $\vartheta(P_{n'-1}) < \chi \leq \vartheta(P_{n'})$ . Namely, we look for the shortest path  $P_{n'}$  such that Lemma 6 can be used to obtain a linear embedding of  $P_{n'}$  with the desired number of crossings. This is given by  $n' = \lceil \frac{5 + \sqrt{25 - 4(6 - 2\chi)}}{2} \rceil$ .

We then execute Algorithm  $\chi$ -Spiral to construct a one-page linear embedding  $\Gamma'$  with  $\chi$  crossings of path  $P_{n'}$ . Once the one-page linear embedding  $\Gamma'$  of  $P_{n'}$  has been computed, we augment  $\Gamma'$  by adding to it a path  $P_{n''}$  with  $n'' = n - n'$  vertices on the same page of  $P_{n'}$  just besides vertex  $v_1$ , positioning its vertices consecutively say before  $v_1$  and joining its last vertex with  $v_1$ . Let  $\Gamma$  be the obtained linear embedding. We have that the edges of  $P_{n''}$  do not cross any edge, hence the total number of crossings of  $\Gamma$  is the same of  $\Gamma'$ . As an example, the linear layout computed for path  $P_9$  and  $\chi = 8$  is shown in Fig. 8c. In this case  $n' = 7$ .

The  $O(n)$  time bound easily descends from the description of the algorithm. ◀

## 5 Point-set Embeddings with Prescribed Edge Crossings

In this section we address the general problem of computing a point-set embedding of a graph  $G$  with prescribed edge crossings and  $O(1)$  curve complexity. The following lemma extends and refines ideas described in the literature for crossing-free point-set embeddings, [25, 39].

► **Lemma 8.** *Let  $G$  be a graph with  $n$  vertices and  $m$  edges. Let  $S$  be a set of  $n$  distinct points in the plane. If  $G$  admits a topological linear embedding with  $\chi$  edge crossings and at most  $\sigma$  spine traversals per edge, then the following hold:*

- *$G$  admits a RAC point-set embedding on  $S$  with  $\chi$  edge crossings and at most  $3(\sigma + 1)$  curve complexity, which can be computed in  $O(m\sigma + n \log n)$  time.*
- *$G$  admits a point-set embedding on  $S$  with  $\chi$  edge crossings and at most  $2\sigma + 1$  curve complexity, which can be computed in  $O(m\sigma + n \log n)$  time.*

**Proof sketch.** We assume, w.l.o.g., that no two points of  $S$  have the same  $x$ -coordinate (this is in fact always the case by suitably rotating the plane). Let  $L_{top}$  be a horizontal line above the point of  $S$  with the largest  $y$ -coordinate and  $L_{bottom}$  be a horizontal line below the point of  $S$  with the smallest  $y$ -coordinate. Let  $\mathcal{S}$  be the strip of plane between  $L_{top}$  and  $L_{bottom}$ . Project the points of  $S$  onto the  $x$ -axis and order them by increasing  $x$ -coordinate. Let  $\Gamma$  be a topological linear layout of  $G$  whose spine is the  $x$ -axis and whose vertices are the projection points of  $S$  onto the spine. For each spine crossing of  $\Gamma$  add a dummy point to  $S$  that has the same  $x$ -coordinate of the spine crossing and is in the interior of  $\mathcal{S}$ . Let  $S'$  be the set of points that includes all points of  $S$  and all dummy points corresponding to the spine crossings of the edges of  $\Gamma$ . In this sketch we show how to construct a RAC point-set embedding, see the appendix for the complete proof.

Since no two points of  $S'$  have the same  $x$ -coordinate, we can define around each point  $s \in S'$  a vertical strip  $\tau_s$  whose interior contains  $s$  and such that no other element of  $S'$  is in the interior or on the boundary of  $\tau_s$ ; we call  $\tau_s$  the *safe strip of  $s$* .

Let  $u$  be a vertex of  $\Gamma$  and let  $s \in S'$  be the point of  $S'$  having the same  $x$ -coordinate as  $u$ . Let  $deg_{top}(u)$  be the number of edges of  $\Gamma$  incident to  $u$  in the top page and let  $deg_{bottom}(u)$  be the number of edges incident to  $u$  in the bottom page. We place  $deg_{bottom}(u)$  dummy points in the interior of  $\tau_s \cap L_{bottom}$  and  $deg_{top}(u)$  dummy points in the interior of  $\tau_s \cap L_{top}$ . We then connect  $s$  to each such dummy points. Each segment connecting  $s$  to a dummy point along  $\tau_s \cap L_{bottom}$  is a *bottom stub* of  $s$ . Each segment connecting  $s$  to a dummy point along  $\tau_s \cap L_{top}$  is a *top stub* of  $s$ . Let  $a$  be a spine crossing along an edge of  $\Gamma$  and let  $s \in S'$  be the point of  $S'$  having the same  $x$ -coordinate as  $a$ . The *top stub* of  $s$  is the vertical segment from  $s$  to the projection of  $s$  onto  $L_{top}$ . The *bottom stub* of  $s$  is the vertical segment from  $s$  to the projection of  $s$  onto  $L_{bottom}$ . The projection point of  $s$  to  $L_{top}$  ( $L_{bottom}$ ) is referred to as the *endpoint of the top (bottom) stub* of  $s$ .

We are now ready to compute a RAC point-set embedding of  $G$ . Every vertex or spine crossing of  $\Gamma$  is mapped to the point of  $S'$  with the same  $x$ -coordinate. For every vertex  $u \in \Gamma$  we order its incident edges in the bottom page from left to right; similarly we order from left to right its incident edges in the top page. Let  $s \in S'$  be the point onto which we map  $u$ : We order both its bottom stubs and its top stubs from left to right. Let  $(u, v)$  be the  $i$ -th edge of  $\Gamma$  incident to  $u$  (either in the bottom or top page of  $\Gamma$ ) and the  $j$ -th edge incident to  $v$  (either in the bottom or top page of  $\Gamma$ ).

Let  $(a, b)$  be a leg of  $(u, v)$  in  $\Gamma$ . Let  $s \in S'$  be the point to which  $a$  is mapped and let  $s' \in S'$  be the point to which  $b$  is mapped. Without loss of generality, we assume that  $a \prec b$  (the case where  $a \succ b$  is handled similarly). If  $a$  and  $b$  are both spine crossings and  $(a, b)$  is in the top (bottom) page of  $\Gamma$ , we connect the endpoint  $p$  of the top (bottom) stub of  $s$  to the endpoint  $p'$  of the top (bottom) stub of  $s'$  with a polyline consisting of two segments: the segment incident to  $p$  has slope  $+1$  ( $-1$ ), the segment incident to  $p'$  has slope  $-1$  ( $+1$ ). Note that the leg  $(a, b)$  is mapped to a polyline with three bends: One at  $p$ , another at  $p'$  and a third one where the two segments with opposite slopes meet. The case where either  $a = u$  and/or  $b = v$  is treated similarly; the only difference being that the polyline representing

the leg  $(a, b)$  in the point-set embedding must be incident to either the  $i$ -th stub of the point representing  $u$  or to the  $j$ -th stub of the point representing  $v$ . In this case  $(a, b)$  is also represented by a polyline having 3 bends. After every leg of  $\Gamma$  has been drawn in the point-set embedding by means of the above procedure, all dummy points corresponding to the spine crossings of  $\Gamma$  are removed from  $S'$ . By construction, every edge of  $\Gamma$  that crosses the spine  $k$  times (and consists of  $k + 1$  legs) is represented in the point-set embedding as a polyline having  $3(k + 1)$  bends. Therefore, if  $\Gamma$  has at most  $\sigma$  spine crossings per edge, the curve complexity of the point-set embedding is  $3(\sigma + 1)$ .

By construction, the stubs are in the interior of the safe strips and the safe strips do not overlap with each other. This implies that the point-set embedding has no edge crossing in the interior of  $\mathcal{S}$ . Also, note that the safe regions follow the left to right order of the points around which they are defined: This order is consistent with the left to right order of the vertices and spine crossings along the spine of  $\Gamma$ . It follows that a crossing occurs in the point-set embedding if and only if we have four points  $s_0, s_1, s_2, s_3$  in  $S'$  with  $x(s_0) < x(s_1) < x(s_2) < x(s_3)$  which correspond to four vertices or spine crossings  $u_0, u_1, u_2, u_3$ , respectively, such that in  $\Gamma$  we have an arc of circumference with diameter  $u_0u_2$ , another arc of circumference with diameter  $u_1u_3$ , such that both legs are in the same page and  $u_0 \prec u_1 \prec u_2 \prec u_3$  in the linear order. This implies that the point-set embedding of  $G$  onto  $S$  has the same number  $\chi$  of edge crossings as the topological linear embedding of  $G$ .

Since all segments along the polylines outside  $\mathcal{S}$  have either slope  $+1$  or slope  $-1$  and since any crossing occurs outside  $\mathcal{S}$ , we also have that the edge crossings form  $\frac{\pi}{2}$  angles and hence the point-set embedding is a RAC drawing.

Concerning the time complexity, the procedure in this sketch first sorts the points of  $S$  by increasing  $x$ -coordinate, then it maps the vertices of  $G$  to the points of  $S$  and then it processes an edge at a time. It spends constant time to draw any leg. It follows that all edges of the point-set embedding are processed in  $O(m\sigma)$ -time, which leads to a computational complexity of  $O(m\sigma + n \log n)$  to compute a (RAC) point-set embedding of  $G$ . ◀

Thm. 5, Lemma 7, and Lemma 8 imply that any tree (resp. path) admits a RAC point-set embedding with a prescribed number of crossings and curve complexity 9 (resp. 3). It can be proved that if the points are collinear, the curve complexity of the RAC point-set embedding can be reduced to 6 (see Fig. 9 for an example and the appendix for details). If we drop the constraint that the edge crossings form right angles, we obtain smaller curve complexities by means of the same Lemmas, as it is summarized in the following theorem.

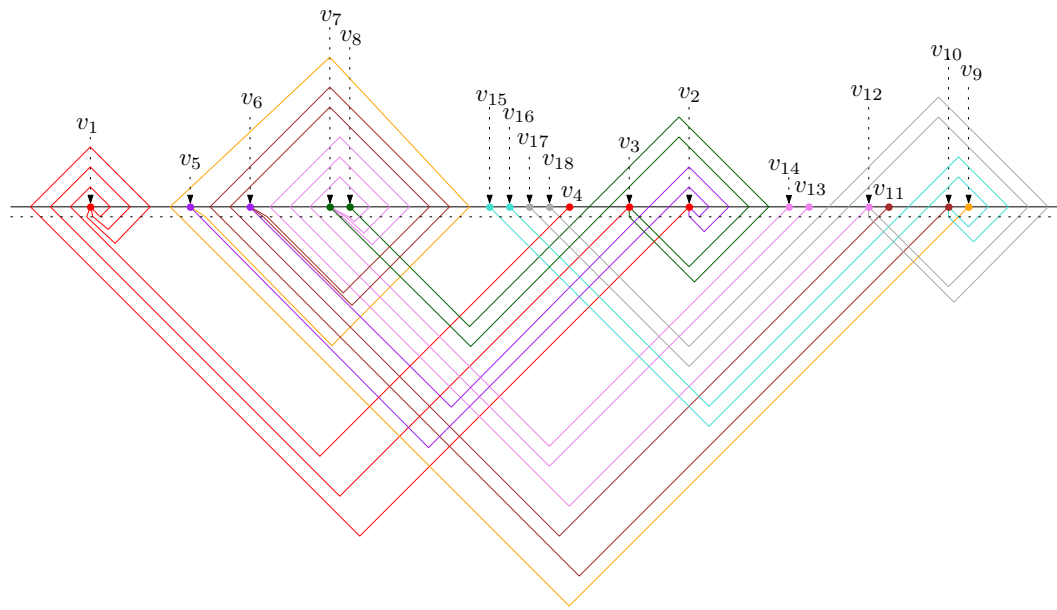
► **Theorem 9.** *Let  $T$  be a tree with  $n$  vertices, let  $\chi \in [0, \vartheta(T)]$ , and let  $S$  be a set of  $n$  distinct points in the plane. The following hold:*

- *We can compute in  $O(n^2)$ -time a point-set embedding of  $T$  on  $S$  with  $\chi$  edge crossings and curve complexity 9 if the drawing is RAC and 5 otherwise. Also, if  $S$  is a set of collinear points, the curve complexity of the RAC point-set embedding reduces to 6.*
- *If  $T$  is a path, we can compute in  $O(n \log n)$ -time a point-set embedding of  $T$  on  $S$  with  $\chi$  edge crossings and curve complexity 3 if the drawing is RAC and 1 otherwise.*

## 6 Final Remarks and Open Problems

Graph Drawing is characterized by a range of computational aesthetic metrics [12, 53]. To investigate the impact of each metric on the readability it is useful to produce different drawings of the same graph where all metrics but one remain fixed [42]. In this context, we compute drawings of trees with any number of crossings (which is probably the most studied





■ **Figure 9** A RAC point-set embedding on collinear points of the tree of Fig. 2a.

aesthetic), while guaranteeing constant curve complexity, and constraining the vertices at given locations.

We conclude with several open problems.

1. Is there an  $o(n^2)$ -time algorithm to compute a point-set embedding of a tree with  $\chi$  crossings and constant curve complexity? This question is interesting even for binary trees.
2. Is it possible to compute RAC point-set embeddings of trees with any number of edge crossings and curve complexity smaller than 9?
3. A seminal paper by Pach and Wenger studies the point-set embeddability without crossings when the mapping between vertices and points is part of the input [45, 46]. It would be interesting to study the question of the present paper also in such a setting.
4. Further, it would be interesting to extend our investigation to graph classes other than trees.

Finally, we believe that examining well-known theoretical problems from an unconventional perspective (see, e.g., [4]) can yield new and interesting insights.

## References

- 1 Patrizio Angelini, Luca Cittadini, Giuseppe Di Battista, Walter Didimo, Fabrizio Frati, Michael Kaufmann, and Antonios Symvonis. On the perspectives opened by right angle crossing drawings. In David Eppstein and Emden R. Gansner, editors, *Graph Drawing, 17th International Symposium, GD 2009, Chicago, IL, USA, September 22-25, 2009. Revised Papers*, volume 5849 of *Lecture Notes in Computer Science*, pages 21–32. Springer, 2009. doi:10.1007/978-3-642-11805-0\_5.
- 2 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.
- 3 Melanie Badent, Emilio Di Giacomo, and Giuseppe Liotta. Drawing colored graphs on colored points. *Theor. Comput. Sci.*, 408(2-3):129–142, 2008. doi:10.1016/J.TCS.2008.08.004.

- 4 Giuseppe Di Battista, Fabrizio Frati, and Maurizio Patrignani. On embedding a graph in the grid with the maximum number of bends and other bad features. *Theory Comput. Syst.*, 44(2):143–159, 2009. doi:10.1007/S00224-008-9115-0.
- 5 Giuseppe Di Battista, Giuseppe Liotta, Maurizio Patrignani, Antonios Symvonis, and Ioannis G. Tollis. Tangling and untangling trees on point-sets. Tech. Report arXiv:2508.18535, Cornell University, 2025. doi:10.48550/arXiv.2508.18535.
- 6 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.
- 7 Therese Biedl and Martin Vatshelle. The point-set embeddability problem for plane graphs. In *Proceedings of the twenty-eighth annual symposium on Computational geometry*, pages 41–50, 2012. doi:10.1145/2261250.2261257.
- 8 Prosenjit Bose. On embedding an outer-planar graph in a point set. In Giuseppe Di Battista, editor, *Graph Drawing, 5th International Symposium, GD '97, Rome, Italy, September 18-20, 1997, Proceedings*, volume 1353 of *Lecture Notes in Computer Science*, pages 25–36. Springer, 1997. doi:10.1007/3-540-63938-1\_47.
- 9 Prosenjit Bose. On embedding an outer-planar graph in a point set. *Comput. Geom.*, 23(3):303–312, 2002. doi:10.1016/S0925-7721(01)00069-4.
- 10 Prosenjit Bose, Michael McAllister, and Jack Snoeyink. Optimal algorithms to embed trees in a point set. In Franz-Josef Brandenburg, editor, *Graph Drawing, GD '95*, volume 1027 of *LNCS*, pages 64–75. Springer, 1995. doi:10.1007/BFB0021791.
- 11 Prosenjit Bose, Michael McAllister, and Jack Snoeyink. Optimal algorithms to embed trees in a point set. *J. Graph Algorithms Appl.*, 1(2):1–15, 1997. doi:10.7155/JGAA.00002.
- 12 Michael Burch, Weidong Huang, Mathew Wakefield, Helen C. Purchase, Daniel Weiskopf, and Jie Hua. The state of the art in empirical user evaluation of graph visualizations. *IEEE Access*, 9:4173–4198, 2021. doi:10.1109/ACCESS.2020.3047616.
- 13 Sergio Cabello. Planar embeddability of the vertices of a graph using a fixed point set is NP-hard. *Journal of Graph Algorithms and Applications*, 10(2):353–363, 2006. doi:10.7155/JGAA.00132.
- 14 Markus Chimani, Stefan Felsner, Stephen Kobourov, Torsten Ueckerdt, Pavel Valtr, and Alexander Wolff. On the maximum crossing number. *Journal of Graph Algorithms and Applications*, 22(1):67–87, March 2018. doi:10.7155/jgaa.00458.
- 15 Giuseppe Di Battista, Peter Eades, Roberto Tamassia, and Ioannis G. Tollis. *Graph Drawing: Algorithms for the Visualization of Graphs*. Prentice-Hall, 1999.
- 16 Emilio Di Giacomo, Walter Didimo, Peter Eades, and Giuseppe Liotta. 2-layer right angle crossing drawings. *Algorithmica*, 68(4):954–997, 2014. doi:10.1007/S00453-012-9706-7.
- 17 Walter Didimo. Right angle crossing drawings of graphs. In Seok-Hee Hong and Takeshi Tokuyama, editors, *Beyond Planar Graphs, Communications of NII Shonan Meetings*, pages 149–169. Springer, 2020. doi:10.1007/978-981-15-6533-5\_9.
- 18 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.
- 19 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.
- 20 Walter Didimo, Giuseppe Liotta, Saeed Mehrabi, and Fabrizio Montecchiani. 1-bend RAC drawings of 1-planar graphs. In Yifan Hu and Martin Nöllenburg, editors, *Graph Drawing and Network Visualization - GD 2016*, volume 9801 of *LNCS*, pages 335–343. Springer, 2016. doi:10.1007/978-3-319-50106-2\_26.
- 21 Walter Didimo, Giuseppe Liotta, and Fabrizio Montecchiani. A survey on graph drawing beyond planarity. *ACM Comput. Surv.*, 52(1), February 2019. doi:10.1145/3301281.
- 22 Peter Eades and Giuseppe Liotta. Right angle crossing graphs and 1-planarity. In Marc J. van Kreveld and Bettina Speckmann, editors, *Graph Drawing - GD 2011*, volume 7034 of *LNCS*, pages 148–153. Springer, 2011. doi:10.1007/978-3-642-25878-7\_15.

- 23 Peter Eades and Giuseppe Liotta. Right angle crossing graphs and 1-planarity. *Discret. Appl. Math.*, 161(7-8):961–969, 2013. doi:10.1016/J.DAM.2012.11.019.
- 24 P. Erdős. On sets of distances of  $n$  points. *The American Mathematical Monthly*, 53(5):248–250, 1946. doi:10.2307/2305092.
- 25 Martin Fink, Jan-Henrik Haunert, Tamara Mchedlidze, Joachim Spoerhase, and Alexander Wolff. Drawing graphs with vertices at specified positions and crossings at large angles. In Md. Saidur Rahman and Shin-Ichi Nakano, editors, *WALCOM: Algorithms and Computation - WALCOM 2012*, volume 7157 of *LNCS*, pages 186–197. Springer, 2012. doi:10.1007/978-3-642-28076-4\_19.
- 26 Radoslav Fulek and János Pach. Thrackles: An improved upper bound. In Fabrizio Frati and Kwan-Liu Ma, editors, *Graph Drawing and Network Visualization - GD 2017*, volume 10692 of *LNCS*, pages 160–166. Springer, 2017. doi:10.1007/978-3-319-73915-1\_14.
- 27 Radoslav Fulek and János Pach. Thrackles: An improved upper bound. *Discrete Applied Mathematics*, 259:226–231, 2019. doi:10.1016/j.dam.2018.12.025.
- 28 W. H. Furry and D. J. Kleitman. Maximal rectilinear crossing of cycles. *Studies in Applied Mathematics*, 56(2):159–167, 1977. doi:10.1002/sapm1977562159.
- 29 Emilio Di Giacomo, Walter Didimo, Giuseppe Liotta, Henk Meijer, Francesco Trotta, and Stephen K. Wismath.  $k$ -colored point-set embeddability of outerplanar graphs. In Michael Kaufmann and Dorothea Wagner, editors, *Graph Drawing, GD 2006*, volume 4372 of *LNCS*, pages 318–329. Springer, 2006. doi:10.1007/978-3-540-70904-6\_31.
- 30 Emilio Di Giacomo, Walter Didimo, Giuseppe Liotta, Henk Meijer, Francesco Trotta, and Stephen K. Wismath.  $k$ -colored point-set embeddability of outerplanar graphs. *J. Graph Algorithms Appl.*, 12(1):29–49, 2008. doi:10.7155/JGAA.00158.
- 31 Emilio Di Giacomo, Leszek Gasieniec, Giuseppe Liotta, and Alfredo Navarra. Colored point-set embeddings of acyclic graphs. In Fabrizio Frati and Kwan-Liu Ma, editors, *Graph Drawing and Network Visualization - GD 2017*, volume 10692 of *LNCS*, pages 413–425. Springer, 2017. doi:10.1007/978-3-319-73915-1\_32.
- 32 Emilio Di Giacomo, Leszek Gasieniec, Giuseppe Liotta, and Alfredo Navarra. On the curve complexity of 3-colored point-set embeddings. *Theor. Comput. Sci.*, 846:114–140, 2020. doi:10.1016/J.TCS.2020.09.027.
- 33 Emilio Di Giacomo, Giuseppe Liotta, and Francesco Trotta. Drawing colored graphs with constrained vertex positions and few bends per edge. In Seok-Hee Hong, Takao Nishizeki, and Wu Quan, editors, *Graph Drawing, GD 2007*, volume 4875 of *LNCS*, pages 315–326. Springer, 2007. doi:10.1007/978-3-540-77537-9\_31.
- 34 Emilio Di Giacomo, Giuseppe Liotta, and Francesco Trotta. Drawing colored graphs with constrained vertex positions and few bends per edge. *Algorithmica*, 57(4):796–818, 2010. doi:10.1007/S00453-008-9255-2.
- 35 Peter Gritzmann, Bojan Mohar, János Pach, and Richard Pollack. Embedding a planar triangulation with vertices at specified points. *The American Mathematical Monthly*, 98(2):165, 1991.
- 36 Heiko Harborth. Drawings of the cycle graph. *Congressus Numerantium*, 66:15–22, January 1988.
- 37 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.
- 38 Michael Kaufmann and Dorothea Wagner, editors. *Drawing Graphs, Methods and Models (the book grow out of a Dagstuhl Seminar, April 1999)*, volume 2025 of *Lecture Notes in Computer Science*. Springer, 2001. doi:10.1007/3-540-44969-8.
- 39 Michael Kaufmann and Roland 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.
- 40 Michael Kaufmann and Roland Wiese. Maintaining the mental map for circular drawings. In Stephen G. Kobourov and Michael T. Goodrich, editors, *Graph Drawing, GD 2002*, volume 2528 of *LNCS*, pages 12–22. Springer, 2002. doi:10.1007/3-540-36151-0\_2.

- 41 L. Lovász, J. Pach, and M. Szegedy. On Conway's thrackle conjecture. *Discrete and Computational Geometry*, 18(4):369–376, 1997. doi:10.1007/PL00009322.
- 42 Gavin J. Mooney, Helen C. Purchase, Michael Wybrow, and Stephen G. Kobourov. The multi-dimensional landscape of graph drawing metrics. In Jorji Nonaka, editor, *Proceedings - 2024 IEEE 17th Pacific Visualization Conference, PacificVis 2024*, pages 122–131, 2024. doi:10.1109/PacificVis60374.2024.00022.
- 43 Rahnuna Islam Nishat, Debajyoti Mondal, and Md. Saidur Rahman. Point-set embeddings of plane 3-trees - (extended abstract). In Ulrik Brandes and Sabine Cornelsen, editors, *Graph Drawing - GD 2010*, volume 6502 of *LNCS*, pages 317–328. Springer, 2010. doi:10.1007/978-3-642-18469-7\_29.
- 44 Rahnuna Islam Nishat, Debajyoti Mondal, and Md Saidur Rahman. Point-set embeddings of plane 3-trees. *Computational Geometry*, 45(3):88–98, 2012. doi:10.1016/J.COMGEO.2011.09.002.
- 45 János Pach and Rephael Wenger. Embedding planar graphs at fixed vertex locations. In Sue Whitesides, editor, *Graph Drawing, GD'98*, volume 1547 of *LNCS*, pages 263–274. Springer, 1998. doi:10.1007/3-540-37623-2\_20.
- 46 János Pach and Rephael Wenger. Embedding planar graphs at fixed vertex locations. *Graphs Comb.*, 17(4):717–728, 2001. doi:10.1007/PL00007258.
- 47 János Pach and Ethan Sterling. Conway's conjecture for monotone thrackles. *The American Mathematical Monthly*, 118(6):544–548, 2011. doi:10.4169/amer.math.monthly.118.06.544.
- 48 B. L. Piazza, R. D. Ringeisen, and S. K. Stueckle. Properties of non-minimum crossings for some classes of graphs. In Yousef Alavi, G. Chartrand, O. R. Oellermann, and A. J. Schwenk, editors, *Graph Theory, Combinatorics, and Applications: Proceedings of the Sixth Quadrennial International Conference on the Theory and Applications of Graphs, Kalamazoo, Western Michigan University, 1988, Volume 2*, pages 975–989. Wiley-Interscience, New York, March 1991.
- 49 Marcus Schaefer. The graph crossing number and its variants: A survey. *The Electronic Journal of Combinatorics*, 2013. Version May 17, 2024. doi:10.37236/2713.
- 50 Roberto Tamassia, editor. *Handbook on Graph Drawing and Visualization*. Chapman and Hall/CRC, 2013. URL: <https://www.crcpress.com/Handbook-of-Graph-Drawing-and-Visualization/Tamassia/9781584884125>.
- 51 D.R. Woodall. Thrackles and deadlock. In D. J. A. Welsh, editor, *Combinatorial Mathematics and its Applications, Proc. 1969 Oxford Combinatorial Conference*, pages 335–347. Academic Press, 2011.
- 52 Yian Xu. A new upper bound for Conway's thrackles. *Applied Mathematics and Computation*, 389:125573, 2021. doi:10.1016/j.amc.2020.125573.
- 53 Vahan Yoghoudjian, Daniel Archambault, Stephan Diehl, Tim Dwyer, Karsten Klein, Helen C. Purchase, and Hsiang-Yun Wu. Exploring the limits of complexity: A survey of empirical studies on graph visualisation. *Vis. Informatics*, 2(4):264–282, 2018. doi:10.1016/J.VISINF.2018.12.006.