DIEI Dipartimento di Ingegneria Elettronica e dell'informazione

RT 005-06

Drawing Colored Graphs on Colored Points

M. Badent[1], *E. Di Giacomo*[2], *G. Liotta*[2]

1 University of Konstanz
2 Università di Perugia

5 December 2006

Via G. Duranti 93, 06125 Perugia, Tel.: 0755853653, Fax: 0755853654,
Web: www.diei.unipg.it, E-mail: segreteria@diei.unipg.it

Drawing Colored Graphs on Colored Points*

Melanie Badent† Emilio Di Giacomo‡ Giuseppe Liotta‡

Abstract

Let G be a planar graph with n vertices whose vertex set is partitioned into subsets $V_0, \ldots, V_{k-1}$ for some positive integer $1 \leq k \leq n$ and let S be a set of n distinct points in the plane partitioned into subsets $S_0, \ldots, S_{k-1}$ with $|V_i| = |S_i|$ $(0 \leq i \leq k-1)$. This paper studies the problem of computing a crossing-free drawing of G such that each vertex of V_i is mapped to a distinct point of S_i. Lower and upper bounds on the number of bends per edge are proved for any $2 \leq k \leq n$. As a special case, we improve the upper and lower bounds presented in a paper by Pach and Wenger for $k = n$ [*Graphs and Combinatorics* (2001), 17:717–728].

1 Introduction

Let G be a planar graph with n vertices whose vertex set is partitioned into subsets $V_0, \ldots, V_{k-1}$ for some positive integer $1 \leq k \leq n$ and let S be a set of n distinct points in the plane partitioned into subsets $S_0, \ldots, S_{k-1}$ with $|V_i| = |S_i|$ $(0 \leq i \leq k-1)$. We say that each index i is a *color*, G is a *k-colored planar graph*, and S is a *k-colored set of points compatible with G*. This paper studies the problem of computing a *k-colored point-set embedding of G on S*, i.e. a crossing-free drawing of G such that each vertex of V_i is mapped to a distinct point of S_i.

Computing k-colored point-set embeddings of k-colored planar graphs has applications in graph drawing, where the *semantic constraints* for the vertices of a graph G define the placement that these vertices must have in a readable visualization of G (see, e.g., [6, 15, 18]). For example, in the context of data base systems design some particularly relevant entities of an ER schema may be required to be drawn in the center and/or along the boundary of the diagram (see, e.g., [19]); in social network analysis, a typical technique to visualize and navigate large networks is to group the vertices into clusters and to draw the vertices of a same cluster close with each other and relatively far from those of other clusters (see, e.g., [5]). A natural way of modelling these types of semantic constraints is to color a (sub)set of the vertices of the input graph and to specify a set of locations having the same color for their placement in the drawing.

As a result, the problem of computing k-colored point-set embeddings of k-colored planar graphs has received considerable interest in the computational geometry and graph drawing communities, where particular attention has been devoted to the *curve complexity* of the computed drawings, i.e. the maximum number of bends along each edge. Namely, reducing the number of bends along the edges is a fundamental optimization goal when computing aesthetically pleasing drawings of graphs (see, e.g., [6, 15, 18]). Before presenting our results, we briefly review the literature on the subject. Since there is not a unified terminology, we slightly rephrase some of the known results; in what follows, n denotes both the number of vertices of a k-colored planar graph and the number of points of a k-colored set of points compatible with the graph.

Kaufmann and Wiese [16] study the "mono-chromatic version" of the problem, that is they focus on 1-colored point-set embeddings. Given a 1-colored planar graph G (i.e. a planar graph G) and a (1-colored) set

*An extended abstract of this paper will appear in the proceedings of the 10^{th} *Workshop on Algorithms and Data Structures, WADS 2007*. Work partially supported by the MIUR Project "MAINSTREAM: Algorithms for massive information structures and data streams"

†Department of Computer and Information Science, University of Konstanz. melanie.badent@uni-konstanz.de

‡Dipartimento di Ingegneria Elettronica e dell'Informazione, Università degli Studi di Perugia. {digiacomo, liotta}@diei.unipg.it

S of points in the plane they show how to compute a 1-colored point-set embedding of G on S such that the curve complexity is at most two, which is proved to be worst case optimal. Further studies on 1-chromatic point-set embeddings can be found in [3, 4, 10]; these papers are devoted to characterizing which 1-colored planar graphs with n vertices admit 1-colored point-set embeddings of curve complexity zero on any set of n points and to presenting efficient algorithms for the computation of such drawings.

2-colored point-set embeddings are studied in [9] where it is proved that subclasses of outerplanar graphs, including paths, cycles, caterpillars, and wreaths all admit a 2-colored point-set embedding on any 2-colored set of points such that the resulting drawing has constant curve complexity. It is also shown in [9] that there exists a 3-connected 2-colored planar graph G and a 2-colored set of points S such that every 2-colored point-set embedding of G on S has at least one edge requiring $\Omega(n)$ bends. These results are extended in [7], where an $O(n \log n)$-time algorithm is described to compute a 2-colored point-set embedding with constant curve complexity for every 2-colored outerplanar graph; in the same paper, it is also proved that for any positive integer h there exists a 3-colored outerplanar graph G and a 3-colored set of points such that any 3-colored point-set embedding of G on S has at least one edge having more than h bends. Characterizations of families of 2-colored planar graphs which admit a 2-colored point-set embedding having curve complexity zero on any compatible 2-colored set of points can be found in [1, 2, 12, 13, 14].

Key references for the "n-chromatic version" of the problem are the works by Halton [11] and by Pach and Wenger [17]. Halton [11] proves that an n-colored planar graph always admits an n-colored point-set embedding on any n-colored set of points; however, he does not address the problem of optimizing the curve complexity of the computed drawing. About ten years later, Pach and Wenger [17] re-visit the question and show that an n-colored planar graph G always has an n-colored point-set embedding on any n-colored set of points such that each edge of the drawing has at most $120n$ bends; they also give a probabilistic argument to prove that, asymptotically, the upper bound on the curve complexity is tight for a linear number of edges. More precisely, let G be an n-colored planar graph with m independent edges and let S be a set of n points in convex position such that each point is colored at random with one of n distinct colors. Pach and Wenger prove that, almost surely, at least $\frac{m}{20}$ edges of G have at least $\frac{m}{40^3}$ bends on any n-colored point-set embedding of G on S.

The present paper describes a unified approach to the problem of computing k-colored point-set embeddings for $2 \leq k \leq n$. The research is motivated by the following observations: (i) The literature has either focused on very few colors or on the n colors case; in spite of the practical relevance of the problem, little seems to be known about how to draw graphs where the vertices are grouped into $2 \leq k \leq n$ clusters and there are semantic constraints for the placement of these vertices. (ii) The $\Omega(n)$ lower bound on the curve complexity for 2-colored point-set embeddings described in [9] implies that for any $2 \leq k \leq n$ there can be k-colored point-set embeddings which require a linear number of bends per edge. This could lead to the conclusion that in order to compute k-colored point-set embeddings that are optimal in terms of curve complexity one can arbitrarily n-color the input graph, consistently color the input set of points, and then use the drawing algorithm by Pach and Wenger [17]. However, the lower bound of [9] shows $\Omega(n)$ curve complexity for a *constant number* of edges, whereas the drawing technique of Pach and Wenger gives rise to a *linear number* of edges each having a linear number of bends. Hence, the total number of bends in a drawing obtained by the technique of [17] is $O(n^2)$ and it is not known whether there are small values of k for which $o(n^2)$ bends would be always possible. (iii) There is a large gap between the multiplicative constant factors that define the upper and the lower bound of the curve complexity of n-colored point-set embeddings [17]. Since the readability of a drawing of a graph is strongly affected by the number of bends along the edges, it is natural to study whether there exists an algorithm that guarantees curve complexity less than $120n$. Our main results are as follows.

- A lower bound on the curve complexity of k-colored point-set embeddings is presented which establishes that $\Omega(n^2)$ bends may be necessary even for small values of k. Namely, it is shown that for every n such that $n \geq 16$ and for every k such that $2 \leq k \leq n$ there exists a k-colored planar graph G with n vertices and a k-colored set of points S compatible with G such that any k-colored point-set embedding of G on S has $\Omega(n)$ edges each having $\Omega(n)$ bends. This lower bound generalizes the one in [17] for $k = n$ and the one in [9] for $k = 2$. Also, the constant factors of our lower bound for $k = n$ are significantly larger than those in [17].

- An $O(n^2 \log n)$-time algorithm is described that receives as input a k-colored planar graph G ($2 \leq k \leq n$), a k-colored set of points S compatible with G, and computes a k-colored point-set embedding of G on S with curve complexity at most $3n + 2$. This reduces by about forty times the previously known upper bound for $k = n$ [17].

- Motivated by the previously described lower bound, special colorings of the input graph are studied which can guarantee a curve complexity that does not depend on n. Namely, it is shown that if the k-colored planar graph G has $k - 1$ vertices each having a distinct color and $n - k + 1$ vertices of the same color, it is always possible to compute a k-colored point-set embedding whose curve complexity is at most $9k - 1$.

Both the lower and the upper bounds are proved by using a common technique, based on translating the geometric problem into a topological augmentation problem. The upper bounds are based on an algorithm that computes a planar drawing of a graph such that all vertices are collinear, the vertices follow a given left-to-right order, and the edges "ripple only a few times".

The remainder of this paper is organized as follows. Preliminary definitions are in Section 2. The lower bound is described in Section 3. Sections 4, 6, and 7 are devoted to the drawing algorithms and their analysis both in terms of computational complexity and in terms of curve complexity. Conclusions and open problems can be found in Section 8.

2 Preliminaries

A *drawing* of a graph G is a geometric representation of G such that each vertex is a distinct point of the Euclidean plane and each edge is a simple Jordan curve connecting the points which represent its endvertices. A drawing is *planar* if any two edges can only share the points that represent common endvertices. A graph is *planar* if it admits a planar drawing.

Let $G = (V, E)$ be a graph. A *k-coloring* of G is a partition $\{V_0, V_1, \ldots, V_{k-1}\}$ of V where the integers $0, 1, \ldots, k-1$ are called *colors*. In the rest of this section the index i is $0 \leq i \leq k-1$ if not differently specified. For each vertex $v \in V_i$ we denote by $col(v)$ the color i of v. A graph G with a k-coloring is called a *k-colored graph*. Let S be a set of distinct points in the plane. We always assume that the points of S have distinct x-coordinates (this condition can always be satisfied by means of a suitable rotation of the plane). For any point $p \in S$ we denote by $x(p)$ and $y(p)$ the x- and y-coordinates of p, respectively. A *k-coloring* of S is a partition $\{S_0, S_1, \ldots, S_{k-1}\}$ of S. A set S of distinct points in the plane with a k-coloring is called a *k-colored set of points*. For each point $p \in S_i$ $col(p)$ denotes the color i of p. A k-colored set of points S is *compatible with* a k-colored graph G if $|V_i| = |S_i|$ for every i; if G is planar, we say that G has a *k-colored point-set embedding* on S if there exists a planar drawing of G such that: (i) every vertex v is mapped to a distinct point p of S with $col(p) = col(v)$, (ii) each edge e of G is drawn as a polyline λ; a point shared by any two consecutive segments of λ is called a *bend* of e. The *curve complexity* of a drawing is the maximum number of bends per edge. Throughout the paper n denotes the number of vertices of graph and m the number of its edges.

3 Lower Bounds on the Curve Complexity

In this section, we first show that for any integer k such that $3 \leq k \leq n$, the problem of computing k-colored point-set embeddings can require a linear number of edges each having a linear number of bends. Then, we show how this result can be extended to 2-colored point-set embeddings.

The lower bound technique for $3 \leq k \leq n$ is based on a deterministic proof and uses combinatorial arguments. We first describe a 3-colored planar graph with n vertices and a 3-colored set of points compatible with this graph. We then show a property of any 3-colored point-set embedding of this graph on the set of points; we finally describe a topological property of the graph. The union of the two properties gives rise to the lower bound. Since the lower bound for the special case of 2-colored point-set embeddings can be proved by means of the same approach but with slight differences in the constant factors, we just state the result in this section and refer the interested reader to the paper appendix for a detailed proof.

3.1 Diamond Graphs and 3-colored Sets of Points

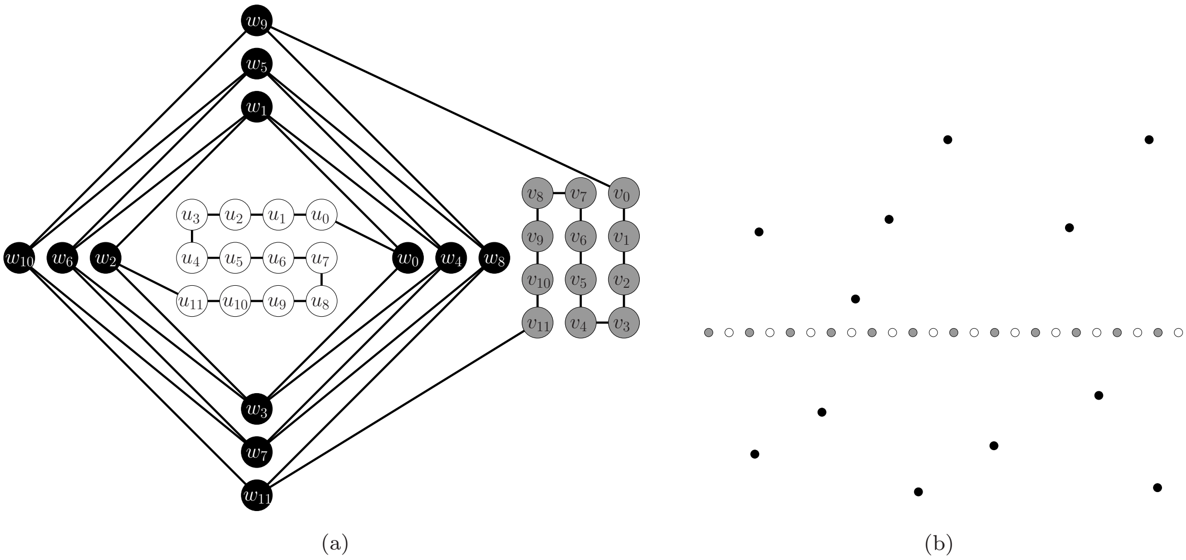


Figure 1: (a) A diamond graph G_n. (b) A 3-colored set of points with an alternating bi-colored sequence compatible with G_n.

A *diamond graph* is a 3-colored planar graph as the one depicted in Figure 1(a). More formally, let $n \geq 12$, let $n'' = (n \mod 12)$ and let $n' = n - n'' = 12h$ for some $h > 0$; a *diamond graph* $G_n = (V, E)$ is defined as follows:

- $V = V_0 \cup V_1 \cup V_2$
- $V_0 = \{v_i \mid 0 \leq i \leq \frac{n'}{3} + \left\lceil \frac{n''}{2} \right\rceil\}$
- $V_1 = \{u_i \mid 0 \leq i \leq \frac{n'}{3} + \left\lfloor \frac{n''}{2} \right\rfloor\}$
- $V_2 = \{w_i \mid 0 \leq i \leq \frac{n'}{3}\}$
- $E = E_0 \cup E_1 \cup E_2 \cup E_3 \cup E_4$
- $E_0 = \{(v_i, v_{i+1}) \mid 0 \leq i \leq \frac{n'}{3} + \left\lceil \frac{n''}{2} \right\rceil - 1\}$
- $E_1 = \{(u_i, u_{i+1}) \mid 0 \leq i \leq \frac{n'}{3} + \left\lfloor \frac{n''}{2} \right\rfloor - 1\}$
- $E_2 = \{(w_i, w_{i+1}),\ (w_{i+1}, w_{i+2}),\ (w_{i+2}, w_{i+3}),\ (w_{i+3}, w_i) \mid 0 \leq i \leq 4h - 1,$ $i \mod 4 = 0\}$
- $E_3 = \{(w_{i+1}, w_{i+4}),\ (w_{i+3}, w_{i+4}),\ (w_{i+1}, w_{i+6}),\ (w_{i+3}, w_{i+6}) \mid 0 \leq i \leq 4h - 5,$ $i \mod 4 = 0\}$

- $E_4 = \{(w_{4h-1}, v_{\frac{n'}{3}+\lceil\frac{n''}{2}\rceil}),\ (w_{4h-3}, v_0),\ (w_0, u_0),\ (w_2, u_{\frac{n'}{3}+\lfloor\frac{n''}{2}\rfloor})\}$

Let $S = S_0 \cup S_1$ be a 2-colored set of points all belonging to a horizontal straight line ℓ; we say that S is an *alternating bi-colored sequence* if $|S_0| = |S_1|$ or $|S_0| = |S_1| + 1$ and no two points of the same color appear consecutively along the line ℓ. A 3-*colored set of points with an alternating bi-colored sequence* is a 3-colored set of points $S = S_0 \cup S_1 \cup S_2$ such that $S' = S_0 \cup S_1$ is an alternating bi-colored sequence with no point of S_2 on ℓ. See Figure 1(b) for an example.

3.2 Bi-colored Paths and Lower Bounds

Let G_n ($n \geq 12$) be the diamond graph with n vertices and let S be a 3-colored set of points with an alternating bi-colored sequence and compatible with G_n. Let Γ_n be a 3-colored point-set embedding of G_n on S. In what follows we shall assume that no bend is represented by a point that belongs to the horizontal straight line ℓ that contains the bi-colored sequence of S. Namely, if a point p representing a bend of an edge of Γ_n is a point of ℓ, we can slightly perturb the drawing so that the drawing remains planar and p is moved either above or below ℓ.

Let $p_0, p_1, \ldots, p_{8h+n''-1}$ be the points of the bi-colored sequence of S ordered according to their x-coordinates. Denote with z_i the vertex of G_n which is mapped to p_i. Notice that z_i and z_{i+1} are not adjacent in Γ_n because one of them belongs to V_0 and the other one belongs to V_1 in G_n. Connect in Γ_n z_i and z_{i+1} with a straight-line segment ($i = 0, \ldots, 8h + n'' - 2$); the obtained path is called *bi-colored path on* Γ_n.

Lemma 1 *Let G_n ($n \geq 12$) be a diamond graph and let S be a 3-colored set of points with an alternating bi-colored sequence such that S is compatible with G_n. Let Γ_n be a 3-colored point-set embedding of G_n on S, let e be an edge of Γ_n, and let Π be the bi-colored path on Γ_n. If Π crosses e b times, then e has at least $b - 1$ bends.*

Proof: Since no bend of Γ_n is on ℓ and no vertex of V_2 is on ℓ then each segment of e can cross the straight line that contains the bi-colored sequence of S at most once. Thus, if e is crossed b times by Π, then it consists of at least b segments. Since at most two endpoints of these segments can be the endvertices of e, it follows that e has at least $b - 1$ bends. □

Lemma 2 *Let G_n ($n \geq 12$) be a diamond graph and let S be a 3-colored set of points with an alternating bi-colored sequence such that S is compatible with G_n. Let Γ_n be a 3-colored point-set embedding of G_n on S and let Π be the bi-colored path on Γ_n. Π crosses at least $\frac{n'}{6} - 1$ edges of Γ_n, where $n' = n - (n \mod 12)$; also, Π crosses each of these edges at least $\frac{n'}{6}$ times.*

Proof: For a planar drawing of G_n and a cycle $C \in G_n$ we say that C *separates* a subset $V' \subset V$ from a subset $V'' \subset V$ if all vertices of V' lie in the interior of the region bounded by C and all vertices of V'' are in the exterior of this region. In every planar drawing of G_n each of the h cycles defined by the edges in the set E_2 separates all vertices in V_0 from all vertices in V_1. Thus every edge of Π must cross these h cycles. Analogously, in every planar drawing of G_n each of the $h - 1$ cycles defined by the edges in the set E_3 separates all vertices in V_0 from all vertices in V_1. Therefore, every edge of Π must also cross these $h - 1$ cycles. The number of edges in Π is $\frac{2n'}{3} + n'' - 1$, where $n'' = n - n' = n \mod 12$, and hence each cycle is crossed $\frac{2n'}{3} + n'' - 1$ times. Since each cycle has four edges, we have that at least $2h - 1 = \frac{n'}{6} - 1$ edges (one per cycle) are crossed at least $\lceil \frac{n'}{6} + \frac{n''}{4} - \frac{1}{4} \rceil \geq \lceil \frac{12h}{6} - \frac{1}{4} \rceil = \lceil 2h - \frac{1}{4} \rceil = 2h = \frac{n'}{6}$ times. □

We are now ready to prove the lower bound.

Theorem 1 *For every $n \geq 12$ and for every $3 \leq k \leq n$ there exists a k-colored planar graph G with n vertices and a k-colored set of points S compatible with G such that any k-colored point-set embedding of G on S has at least $\frac{n'}{6} - 1$ edges each having at least $\frac{n'}{6} - 1$ bends, where $n' = n - (n \mod 12)$.*

Proof: Given any $n \geq 12$ construct a diamond graph G_n and consider a 3-colored set of points S with an alternating bi-colored sequence which is compatible with G_n.

Arbitrarily divide the set of colors $\{0, 1, \ldots, k-1\}$ in three non-empty subsets C_0, C_1 and C_2. Arbitrarily color the vertices of G_n in the set V_i by using the colors in the set C_i ($i = 0, 1, 2$), with the only requirement that each color is used at least once. Analogously, arbitrarily color the points of S in the set S_i by using the colors in the set C_i ($i = 0, 1, 2$) with the only requirement that S remains compatible with G_n. As a result we have a k-colored graph G_n with n vertices and a k-colored set of points S compatible with G_n. Let Γ_n be a k-colored point-set embedding of G_n on S. Let Π be the bi-colored path on Γ_n. By Lemma 2 there are at least $\frac{n'}{6} - 1$ edges of Γ_n that are crossed by Π at least $\frac{n'}{6}$ times. By Lemma 1 each of these edges has at least $\frac{n'}{6} - 1$ bends in Γ_n. □

We can compare the result of Theorem 1 with the known lower bound for $k = n$ [17]. Let G be an n-colored graph with m independent edges and let S be a set of n points in convex position such that each point is colored at random with one of n distinct colors. In [17] it is proved that, almost surely, at least $\frac{m}{20}$ edges of G have at least $\frac{m}{40^3}$ bends on any possible n-colored point-set embedding of G on S. A comparison with the result in Theorem 1 can be easily done by observing that the maximum number of independent edges in a graph with n vertices is at most $n/2$. Also, we remark the argument of Theorem 1 is deterministic and that it can be applied to all values of k such that $3 \leq k \leq n$.

We conclude this section by extending Theorem 1 to the case of 2-colored point-set embeddings. The extension uses the same reasoning illustrated above for three or more colors, but it requires slightly different definitions and gives rise to slightly smaller constant factors. While all details have been moved to the paper appendix, we give here only a brief sketch of the ideas behind this lower bound. Intuitively, a *2-colored diamond graph* can be regarded as a diamond graph where the vertices of set V_1 and V_2 have the same color and the vertices of set V_0 are such that $|V_0| = |V_1| + |V_2|$. Figure 2(a) is an example of a 2-colored diamond graph (see also the appendix for a formal definition of a 2-colored diamond graph); Figure 2(b) is an alternating bi-colored sequence compatible with the graph of Figure 2(a). With the same reasoning illustrated above, the following can be proved (see the appendix for details).

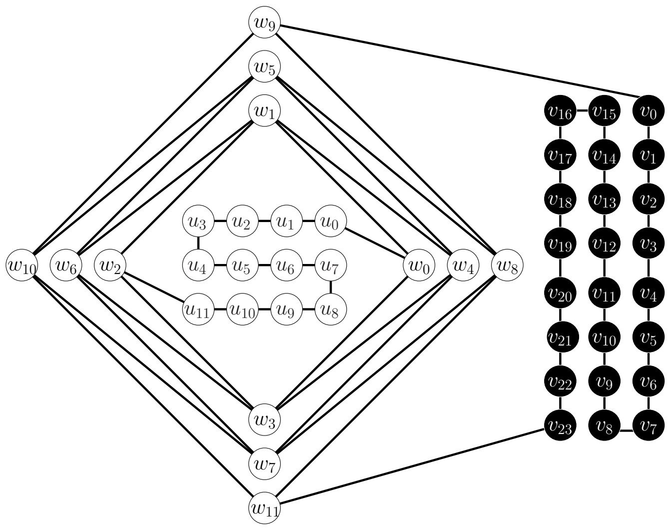
Theorem 2 *For every $n \geq 16$ there exists a 2-colored planar graph G_n with n vertices and a 2-colored set of points S compatible with G_n such that any 2-colored point-set embedding of G_n on S has at least $\frac{n'}{8} - 1$ edges each having at least $\frac{n'}{8} - 1$ bends, where $n' = n - (n \mod 16)$.*

4 Upper Bounds: Overview of the Approach

Theorems 1 and 2 show that, for every $2 \leq k \leq n$, $\Omega(n)$ bends per edge can be required in a k-colored point-set embedding of a k-colored graph G with n vertices. Therefore, a drawing algorithm that is asymptotically optimal in terms of curve complexity for all values of k such that $2 \leq k \leq n$ can be designed as follows: (1) Arbitrarily assign each vertex of G having color i to a distinct point of color i (if there is more than one vertex of G having color i); and (2) Apply the drawing algorithm of Pach and Wenger [17], which computes an n-colored point-set embedding of G whose curve complexity is at most $120n$.

However, since optimizing the number of bends per edge is an important requirement that guarantees the readability of a drawing of a graph [6, 15, 18], we present in the next three sections a new drawing strategy that gives rise to n-colored point-set embeddings with curve complexity at most $3n + 2$. The key idea is to translate the geometric problem into an equivalent topological problem, namely that of computing a Hamiltonian path of a planar graph by suitably augmenting it with dummy edges that do not cross the real edges "too many times". An overview of the content of the next three sections is as follows:

- The notion of *augmenting k-colored Hamiltonian path* for a k-colored planar graph G is introduced (Section 5).
- A theorem that proves that the number of crossings between the edges of an augmenting k-colored Hamiltonian path and the edges of G define an upper bound on the curve complexity of a k-colored point-set embedding of G is proved (Theorem 3).



(a)

(b)

Figure 2: (a) A 2-colored diamond graph G_n. (b) An alternating bi-colored sequence compatible with G_n.

- An algorithm that, for any linear ordering of the vertices of G, computes a planar drawing of G such that all vertices are collinear, the vertices in the drawing follow the given ordering , and each edge can be decomposed into at most three x-monotone curves is presented (Section 6).
- Finally, the above algorithm is exploited to compute a k-colored hamiltonian path on G and then a k-colored point-set embedding such that every edge bends at most $3n + 2$ times. (Section 7).

5 Colored Hamiltonicity

A *k-colored sequence* σ is a linear sequence of (possibly repeated) colors $c_0, c_1, \ldots, c_{n-1}$ such that $0 \leq c_j \leq k-1$ ($0 \leq j \leq n-1$). We say that σ is *compatible with* a k-colored graph G if, for every $0 \leq i \leq k-1$, color i occurs $|V_i|$ times in σ. Let S be a k-colored set of points and let $p_0, p_1, \ldots, p_{n-1}$ be the points of S ordered according to their x-coordinates. We say that S *induces* the k-colored sequence $\sigma = col(p_0), col(p_1), \ldots, col(p_{n-1})$. Figures 3(a) and 3(b) show an example of a 3-colored planar graph and of a 3-colored sequence compatible with it and induced by a 3-colored set of points.

A graph G has a *Hamiltonian path* if it has a simple path that contains all the vertices of G. If G is a k-colored graph and $\sigma = c_0, c_1, \ldots, c_{n-1}$ is a k-colored sequence compatible with G, a *k-colored Hamiltonian path of G consistent with σ* is a Hamiltonian path $v_0, v_1, \ldots, v_{n-1}$ such that $col(v_i) = c_i$ ($0 \leq i \leq n-1$). A k-colored planar graph G can always be augmented to a (not necessarily planar) k-colored graph G' by adding to G a suitable number of dummy edges and such that G' has a k-colored Hamiltonian path $\mathcal{H}'$ consistent with σ and that includes all dummy edges. Figure 3(c) shows an augmentation of the graph of Figure 3(a) such that the augmented (non-planar) graph has a 3-colored Hamiltonian consistent with the sequence of Figure 3(b).

If G' is not planar, we can apply a planarization algorithm (see, e.g., [6]) to G' with the constraint that only crossings between dummy edges and edges of $G - \mathcal{H}'$ are allowed (see Figure 3(d)). Such a planarization algorithm constructs an embedded planar graph G'', called *augmented Hamiltonian form of G*, where each edge crossing is replaced with a dummy vertex, called *division vertex*. By this procedure, an edge e of $\mathcal{H}'$ can be transformed into a path whose internal vertices are division vertices. The subdivision of $\mathcal{H}'$ obtained this way is called an *augmenting k-colored Hamiltonian path of G consistent with σ* and is denoted as $\mathcal{H}''$. If every edge e of G is crossed at most d times in G' (i.e. e is split by at most d division vertices in G''), $\mathcal{H}''$ is said to be an *augmenting k-colored Hamiltonian path of G consistent with σ and inducing at most d division vertices per edge*. Notice that d is the number of division vertices that have been inserted along each edge of G; for example, the path $\mathcal{H}''$ of Figure 3(d) is an augmenting 3-colored Hamiltonian path of the graph of Figure 3(a) consistent with the sequence of Figure 3(b) and inducing one division vertex per edge, because each edge of $\mathcal{H}''$ crosses each edge of G at most once. If G' is planar, then the augmented Hamiltonian form of G is G' and $\mathcal{H}''$ coincides with $\mathcal{H}'$. If both endvertices of $\mathcal{H}''$ are on the external face of the augmented Hamiltonian form of G, then $\mathcal{H}''$ is said to be *external*.

Let v_d be a division vertex for an edge e of G. Since a division vertex corresponds to a crossing between e and an edge of $\mathcal{H}'$, there are four edges incident on v_d in G''; two of them are dummy edges that belong to $\mathcal{H}''$, the other two are two "pieces" of edge e obtained by splitting e with v_d. Let (u, v_d) and (v, v_d) be the latter two edges. We say that v_d is a *flat division vertex* if it is encountered after u and before v while walking along $\mathcal{H}''$; v_d is a *pointy division vertex* otherwise. The following theorem refines and improves a similar result presented in [7]. The algorithm described in its proof is based on the drawing technique of Kaufmann and Wiese [16].

Theorem 3 *Let G be a k-colored planar graph with n vertices, let σ be a k-colored sequence compatible with G, and let $\mathcal{H}$ be an augmenting k-colored Hamiltonian path of G consistent with σ inducing at most d_f flat and d_p pointy division vertices per edge. If $\mathcal{H}$ is external then G admits a k-colored point-set embedding on any set of points that induces σ such that the maximum number of bends along each edge is $d_f + 2d_p + 1$.*

Proof: Let S be a k-colored set of points that induces the k-colored sequence σ. We shall use path $\mathcal{H}$ to construct a k-colored point-set embedding of G on S. Let $\mathcal{H} = w_0, w_1, \ldots, w_{n'-1}$. Path $\mathcal{H}$ contains also the

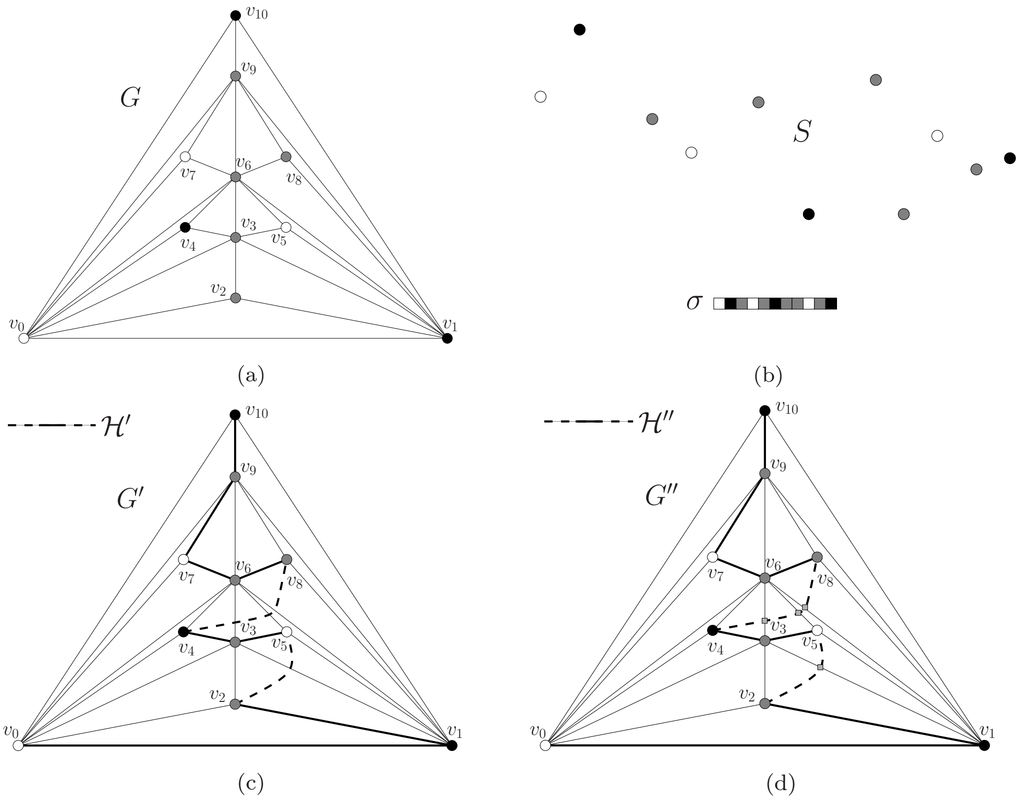


Figure 3: (a) A 3-colored planar graph graph G. (b) A 3-colored set of points S consistent with G and its induced 3-colored sequence σ, compatible with G. (c) An augmentation of G to a (non-planar) 3-colored graph G' that admits a 3-colored Hamiltonian path $\mathcal{H}'$ consistent with σ. Path $\mathcal{H}'$ is highlighted in bold. Dashed edges are dummy edges. (d) A planar graph G'' obtained by applying a planarization algorithm to G'. The path highlighted in bold is an augmenting 3-colored Hamiltonian path $\mathcal{H}''$ of G consistent with σ and inducing at most 1 division vertex per edge.

division vertices, which are not vertices of G. We give these vertices a new color k. In order to draw them we define a new set of points S' by adding a suitable number of points to S, all having color k and placed so that if $q_0, q_1, \ldots, q_{n'-1}$ are the points of S' ordered according to their x-coordinates, then $c(q_j) = c(w_j)$ ($j = 0, \ldots, n'-1$). In the following we denote as G' the augmented Hamiltonian form of G. We can now use the the drawing technique of Kaufmann and Wiese [16] to point set embed G' on S'; for completeness, we recall this technique in the following.

Map each vertex w_j to point q_j ($j = 0, \ldots, n'-1$) in S' and draw the edges of path $\mathcal{H}$ as straight-line segments between their endvertices. Each edge e not in $\mathcal{H}$ is drawn by using two segments, one with slope $s > 0$ and the other with slope $-s$. In order to avoid crossings between e and the edges in $\mathcal{H}$ the slope s is chosen to be greater than the absolute value of the slope of each edge in $\mathcal{H}$. With segments of slope $\pm s$, it is possible to draw each edge e above or below $\mathcal{H}$. Since $\mathcal{H}$ is external, there exists a planar embedding of G' such that w_0 and $w_{n'-1}$ are on the external face. In such an embedding every edge not in $\mathcal{H}$ is either on the left-hand side of $\mathcal{H}$, in which case it is drawn above $\mathcal{H}$, or on the right-hand side of $\mathcal{H}$ when walking from w_0 to $w_{n'-1}$, in which case it is drawn below $\mathcal{H}$.

The resulting drawing is planar except that edges outside $\mathcal{H}$ that are incident on the same vertex may contain overlapping segments. To eliminate overlapping, perturb overlapping edges by decreasing the absolute value of their segment slopes by slightly different amounts. The slope changes are chosen to be small enough to avoid creating edge crossings while preserving the same planar embedding. For details about this rotation see [16].

The drawing obtained by the technique described above is a $(k+1)$-colored point-set embedding of G' on S' with at most one bend per edge. Removing the vertices and edges added to obtain G' from G we have a k-colored point-set embedding of G on S. Consider an edge e of G and suppose that e is split by means of $d_t = d_f + d_p$ division vertices in G'. Then there are $d_t + 1$ edges in G' corresponding to e, each one having at most one bend. As we pointed out above, there are four edges incident on every dummy vertex d; two of them are dummy edges that belong to $\mathcal{H}$, the other two are two "pieces" of the real edge e obtained by splitting e by means of d. After the removal of dummy elements (vertices and edges) only the latter two edges remain in the drawing. Denote them as (u, d) and (v, d). Since these edges are not in $\mathcal{H}$, one of them is above $\mathcal{H}$ and the other one is below $\mathcal{H}$. Thus a segment s_u of (u, d) and a segment s_v of (v, d) are incident on d, one from above and one from below. Since d has only one segment incident from above and only one segment incident from below, the rotation performed to remove overlap does not affect s_u and s_v, which therefore have slope either $+s$ or $-s$. If d is a pointy division vertex then s_u and s_v have different slopes and the removal of d gives an extra bend; if d is a flat division vertex, then s_u and s_v have the same slope and d can be removed without introducing any extra bend. Thus we can have d_p extra bends for an overall curve complexity of $d_t + 1 + d_p = d_f + 2d_p + 1$. □

Based on Theorem 3, we will show our upper bound by proving that for *any* n-colored sequence σ an n-colored planar graph G always admits an augmenting k-colored Hamiltonian path of G consistent with σ such that for each edge $d_f \leq 3n - 3$ and $d_p \leq 2$.

6 Computing Topological Book Embeddings with a Given Linear Ordering

The algorithm to compute an augmenting k-colored Hamiltonian path of G consistent with σ relies on a geometric technique that starts with a topological book embedding of G (a special type of planar drawing where all vertices are aligned, defined in the next paragraphs) and transforms it into a new topological book embedding that respects the given linear ordering for the vertices of G.

A *spine* is an horizontal line. Let ℓ be a spine and let p, q be two points of ℓ. An *arc* is a circular arc passing through the three points p, q, and r, where r is a point of the perpendicular bisector of $\overline{pq}$, at a distance $\frac{d(p,q)}{4}$ from ℓ. The arc can be either in the half-plane above the spine or in the half-plane below the spine; in the first case we say that the arc is in the *top page* of ℓ, otherwise it is in the *bottom page* of ℓ.

Let $G = (V, E)$ be a planar graph. A *topological book embedding* of G is a planar drawing such that all vertices of G are represented as points of a spine ℓ and each edge can be either above the spine, or below the spine, or it can cross the spine. Each crossing between an edge and the spine is called a *spine crossing*. It is also assumed that in a topological book embedding every edge consists of one or more arcs such that no two consecutive arcs are in the same page. An edge e is said to be in the top (bottom) page of the spine if it consists of exactly one arc and this arc is in the top (bottom) page. Figure 4 shows two examples of topological book embeddings.

A *monotone topological book embedding* is a topological book embedding such that each edge crosses the spine at most once. Also, let $e = (u, v)$ be an edge of a monotone topological book embedding that crosses the spine at a point p; e is such that if u precedes v in the left-to-right order along the spine then p is between u and v, the arc with endpoints u and p is in the bottom page, and the arc with endpoints u and v is in the top page. Figure 4(a) is an example of a monotone topological book embedding of a planar graph.

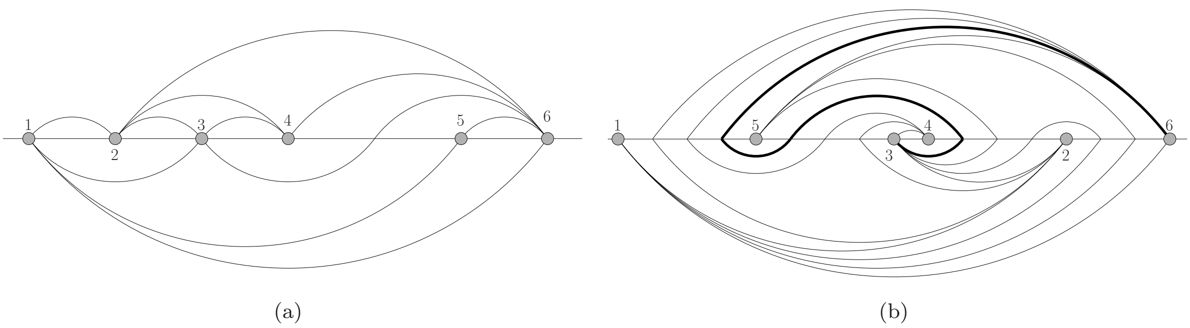


Figure 4: Two topological book embeddings of a planar graph G. (a) A monotone topological book embedding of G. (b) A 3-chain topological book embedding of G. The bold edge consists of three x-monotone chains.

Theorem 4 [8] *Every planar graph admits a monotone topological book embedding. Also, a monotone topological book embedding can be computed in $O(n)$ time, where n is the number of the vertices in the graph.*

Let $e = (u, v)$ be an edge of a topological book embedding. An *x-monotone portion of e* is a portion π_e of e such that every vertical line intersects π_e at most once. An x-monotone portion of e is *maximal* if it is not contained in any other x-monotone portion of e. A maximal x-monotone portion of e is called an *x-monotone chain* of e. We say that a topological book embedding is a *k-chain topological book embedding* if each edge consists of at most k x-monotone chains. Figure 4(b) is an example of a 3-chain monotone topological book embedding of the same graph of Figure 4(a): the bold edge in the drawing consists of three x-monotone chains and all other edges consist of at most two x-monotone chains. Notice that the linear order of the vertices along the spine in Figure 4(b) is different from the one in Figure 4(a).

Before presenting our drawing algorithm to compute a topological book embedding with a given left-to-right order of the vertices along the spine we need to introduce another concept, which generalizes the notion of topological book embedding. Let ℓ and ℓ' be two distinct spines such that ℓ is above ℓ'; ℓ is called *upper spine* and ℓ' is called *lower spine*. A 2-spine drawing Γ^* of G is a (not necessarily planar) drawing such that each vertex of G is represented as a point either of the upper spine or of the lower spine and each edge crosses the spines a finite number of times. More precisely, an edge of a 2-spine drawing can have both endvertices in the same spine, or in different spines. If both endvertices are in the same spine, the edge consists of a sequence of arcs such that any two consecutive arcs are on opposite pages of the spine. If one endvertex is in the upper spine and the other is in the lower spine, then the edge consists of: (i) a (possibly empty) sequence of arcs whose endpoints are in the upper spine, called the *upper sequence* of the edge; (ii) a straight-line segment between the two spines, called the *inter-spine segment* of the edge; (iii) a (possibly empty) sequence of arcs whose endpoints are in the lower spine, called the *lower sequence* of the edge. It is also assumed that any two consecutive arcs of the upper (lower) sequence are on opposite pages of the upper (lower) spine. In what follows, we shall sometimes treat arcs and inter-spine segments in the same way; in these cases we shall use the term *sub-edge* of an edge to mean either an arc or an inter-spine segment of an edge in a 2-spine drawing.

Figure 5 is an example of a planar 2-spine drawing of the same graph of Figure 4(a); $(1, 3)$ is an example of an edge with both endvertices on the same spine. Edge $e = (2, 6)$ in Figure 5 has its endvertices on different spines: The upper sequence is the sequence of arcs of e from p to 6; the straight-line segment $\overline{pq}$ is the inter-spine segment of e; the sequence of arcs of e from q to 2 is the lower sequence of e.

Observe that if all vertices are on the same (upper or lower) spine and if the drawing is planar, a 2-spine drawing of a graph is a topological book embedding of the graph.

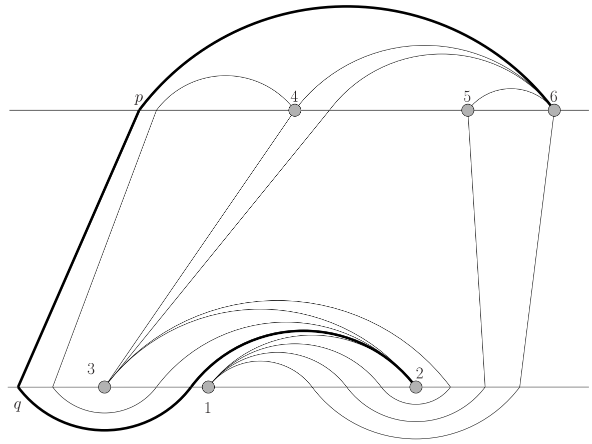


Figure 5: A 2-spine drawing of the graph in Figure 4(a). The bold edge has an upper sequence, an inter-spine segment, and a lower sequence.

6.1 Algorithm LinearOrderDraw

Algorithm LinearOrderDraw receives as input a planar graph G with n vertices and a linear ordering λ of the vertices of G. It produces as output a 3-chain topological book embedding Γ' of G such that the left-to-right order of the vertices along the spine of Γ' is λ. By using Theorem 4, Algorithm LinearOrderDraw computes first a monotone topological book embedding of G, denoted as Γ; then, it transforms Γ into the 3-chain topological book embedding Γ'. Let ℓ be the spine of Γ and let $v_0, \dots, v_{n-1}$ be the vertices of G in the left-to-right order they have along ℓ (note that this order can be different from λ).

A horizontal line below ℓ is chosen as the spine of Γ' and is denoted as ℓ'. Let δ be the distance between the leftmost vertex and the rightmost vertex of Γ along spine ℓ. Choose the distance between ℓ and ℓ' greater than $\sqrt{3}\delta$. Also, choose an interval I on ℓ' of size at most δ. Every vertex v of G has a *source position* $s(v)$ defined by the point along ℓ representing v in Γ and a *target position* $t(v)$ on ℓ' such that $t(v)$ will represent v in Γ'. The target positions are chosen inside interval I in such a way that their left-to-right order corresponds to λ. Also, the endpoints of every arc a that Algorithm LinearOrderDraw will draw either in top or in the bottom page of the lower spine will be points inside interval I. The *trajectory* of vertex v is the straight-line segment $\overline{s(v)t(v)}$ and it is denoted as $\tau(v)$.

Algorithm LinearOrderDraw visits the vertices of Γ in the left-to-right order along ℓ and executes n steps. At each step, a vertex is moved to its target position on ℓ' and a planar 2-spine drawing with upper spine ℓ and lower spine ℓ' of G is computed. More precisely, a sequence $\Gamma_0, \dots, \Gamma_n$ of planar 2-spine drawings with spines ℓ and ℓ' are computed such that Γ_0 coincides with Γ and Γ_n coincides with Γ'. At Step i $(0 \leq i \leq n-1)$, the planar 2-spine drawing Γ_i is transformed into the planar 2-spine drawing Γ_{i+1} by moving v_i to its target position on ℓ'. When moving vertex v_i to its target position, Algorithm LinearOrderDraw maintains the planar embedding of Γ and changes only the shape of those edges incident on v_i and the shape of every edge that is intersected by the trajectory of v_i. Details on how the shapes of these edges are changed are given below.

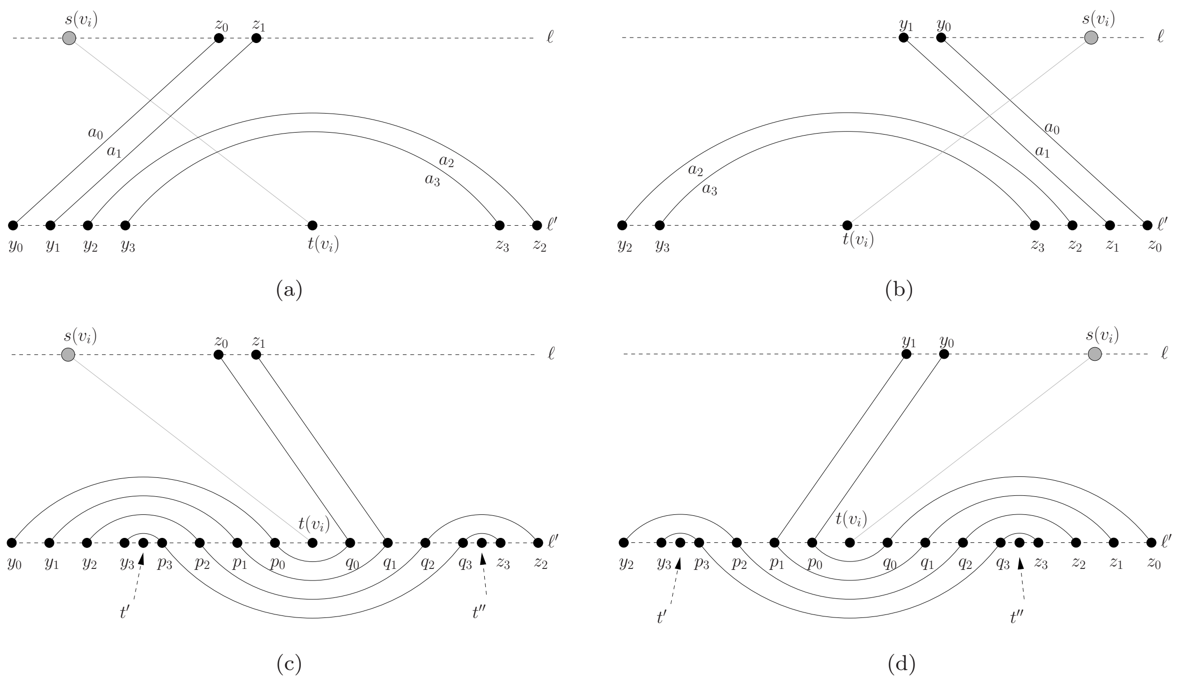


Figure 6: Illustration of Step i of Algorithm LinearOrderDraw: Transformation of the shape of the edges intersected by the trajectory $\tau(v_i)$ of v_i. The trajectory is the light grey segment. (a) and (c) describe the change of the shapes of left inter-spine segments and of arcs in the lower sequence. (b) and (d) describe the change of the shapes of right inter-spine segments and of arcs in the lower sequence.

- **Transformation of the shape of the edges intersected by the trajectory of v_i.** The trajectory $\tau(v_i)$ can intersect both inter-spine segments and arcs of the lower sequence of some edges. Let $a_0, a_1, \ldots, a_{h-1}$ be the sub-edges crossed by $\tau(v_i)$ in the order they are encountered when going from $s(v_i)$ to $t(v_i)$ along $\tau(v_i)$. If a_j is an arc, denote its endpoints on ℓ' as y_j and z_j and assume y_j to the left of z_j. If a_j is an inter-spine segment and the endpoint of a_j that is on ℓ' is to the left of $t(v_i)$ denote this endpoint as y_j, the other one as z_j, and call the inter-spine segment a *left inter-spine segment* (see also Figure 6(a)); if, otherwise, the endpoint of a_j that is on ℓ' is to the right of $t(v_i)$ denote this endpoint as z_j, the other one as y_j, and call the inter-spine segment a *right inter-spine segment* (see also Figure 6(b)).

  Algorithm LinearOrderDraw modifies the shape of the h sub-edges $a_0, a_1, \ldots, a_{h-1}$ intersected by $\tau(v_i)$ as follows. Refer to Figures 6(c) and 6(d). Let t' and t'' be two points of $\ell' \cap I$ such that t', $t(v_i)$ and t'' appear in this left-to-right order along ℓ' and no vertex or spine crossing is between t' and $t(v_i)$ and between $t(v_i)$ and t'' on ℓ'. Choose h points $p_0, p_1, \ldots, p_h$ such that each p_j $(0 \leq j \leq h)$ is between t' and $t(v_i)$ on ℓ' and p_j is to the right of p_{j+1} on ℓ' $(0 \leq j \leq h-1)$. Choose h points $q_0, q_1, \ldots, q_h$ such that each q_j $(0 \leq j \leq h)$ is between $t(v_i)$ and t'' on ℓ' and q_j is to the left of q_{j+1} on ℓ' $(0 \leq j \leq h-1)$. If a_j is an arc it is replaced by: (i) an arc with endpoints y_j and p_j; (ii) an arc with endpoints p_j and q_j; (ii) an arc with endpoints q_j and z_j. If a_j is a left inter-spine segment it is replaced by: (i) an arc with endpoints y_j and p_j; (ii) an arc with endpoints p_j and q_j; (iii) an inter-spine segment with endpoints q_j and z_j (Figure 6(c)). If a_j is a right inter-spine segment it is replaced by: (i) an inter-spine segment with endpoints y_j and p_j; (ii) an arc with endpoints p_j and q_j; (iii) an arc with endpoints q_j and z_j (Figure 6(d)).

- **Transformation of the shape of the edges incident on v_i.** Partition the edges incident on v_i in the drawing Γ into four sets. The set $E_{t,l}$ ($E_{b,l}$) contains the edges $e = (v_j, v_i)$ such that $j < i$ and the arc of e incident on v_i is in the top (bottom) page of the spine ℓ of Γ. Analogously, $E_{t,r}$ ($E_{b,r}$) contains the edges $e = (v_j, v_i)$ such that $i < j$ and the arc of e incident on v_i is in the top (bottom) page of ℓ.

  – Let $e = (v_j, v_i)$ be an edge of $E_{t,l}$ or $E_{b,l}$. Refer to Figure 7. When v_i is moved to its target position, v_j has already been processed and moved to its target position on ℓ' during a previous step of Algorithm LinearOrderDraw because $j < i$ and the algorithm processes the vertices of Γ in a left-to-right order. Hence, when going from v_j to v_i along e in Γ_i we find the (possibly empty) lower sequence σ_l of e, the inter-spine segment a of e, and the (possibly empty) upper sequence σ_u of e. Let x' be the endpoint of a on ℓ'. Replace a and σ_u with an arc whose endpoints are x' and $t(v_i)$.

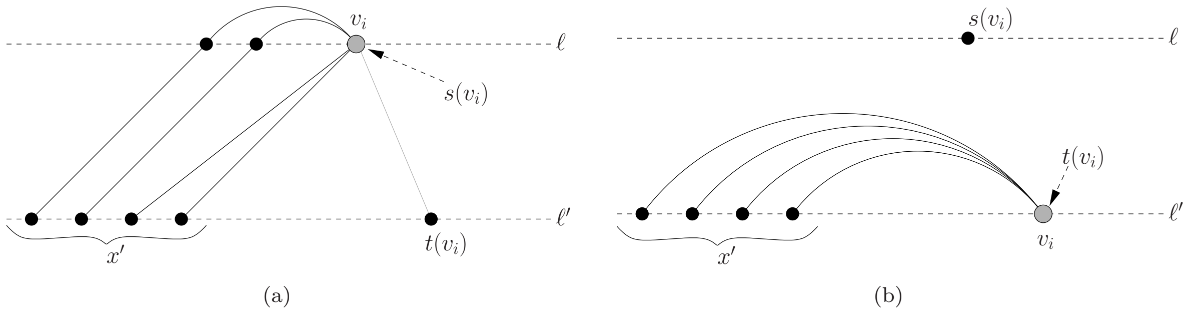


Figure 7: Illustration of Step i of Algorithm LinearOrderDraw: Transformation of the shape of the edges incident on v_i and belonging to $E_{t,l}$ or $E_{b,l}$.

  – Let $e = (v_i, v_j)$ be an edge of $E_{b,r}$. Refer to Figure 8). Edge e is represented in Γ_i as an arc a with endpoints $s(v_i)$ and $s(v_j)$. Arc a is replaced by the straight-line segment $\overline{t(v_i)s(v_j)}$.
  – Let $e_j = (v_i, v_{i_j})$ $(0 \le j \le h-1)$ be the edges of $E_{t,r}$ with $i_j < i_{j+1}$ $(0 \le j < h-1)$. Refer to Figure 8. Let s' be a point on ℓ such that s' is to the right of $s(v_i)$ and no vertex or spine crossing is between $s(v_i)$ and s' on ℓ. Choose h points $p_0, p_1, \ldots, p_{h-1}$ such that each p_j $(0 \le j \le h-1)$ is between $s(v_i)$ and s' on ℓ and p_j is to the left of p_{j+1} along ℓ $(0 \le j < h-1)$. Edge e_j is represented in Γ_i as an arc a_j with endpoints $s(v_i)$ and $s(v_{i_j})(0 \le j \le h-1)$. Arc a_j is replaced by the segment $\overline{t(v_i)p_j}$ and the arc with endpoints p_j and $s(v_{i_j})$.

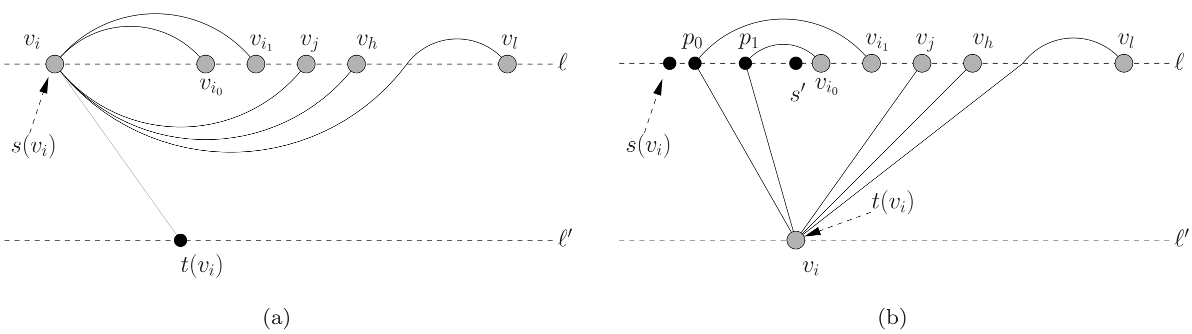


Figure 8: Illustration of Step i of Algorithm LinearOrderDraw: Transformation of the shape of the edges incident on v_i and belonging to $E_{t,r}$ or $E_{b,r}$.

6.2 Analysis of Algorithm LinearOrderDraw

In this section we prove the correctness of Algorithm LinearOrderDraw and analyze its time complexity. As explained in the previous section, Algorithm LinearOrderDraw computes first a monotone topological book embedding Γ_0 and then it executes n steps to transform Γ_0 into a 3-chain topological book embedding. We distinguish each of these n steps with an index i such that $0 \leq i \leq n-1$; recall that Step i computes a drawing denoted as Γ_{i+1}. Also, we shall conventionally denote as Step (-1) the initial step that computes Γ_0. Let a be a sub-edge of a drawing Γ_i that is replaced in Γ_{i+1} by other sub-edges, and let a' be one of these sub-edges. We say that a' *replaces* a; we also say that a' is a *replacing sub-edge of Step* i. We start by proving that the output of each step is a 2-spine drawing.

Lemma 3 *Let Γ_i be the drawing computed by Step $(i-1)$ of Algorithm* LinearOrderDraw *($0 \leq i \leq n-1$). Γ_i is a 2-spine drawing.*

Proof: Step (-1) computes a monotone topological book embedding Γ_0 by using Theorem 4. By definition, a monotone topological book embedding is also a 2-spine drawing.

Assume by induction that the drawing Γ_i computed by Step $(i-1)$ $(1 \leq i \leq n-1)$ is a 2-spine drawing. The vertices of Γ_{i+1} are either points of the lower or of the upper spine by construction. Step i of Algorithm LinearOrderDraw modifies the shape of those edges that are intersected by the trajectory of v_i and of those edges that are incident to v_i. Let e be an edge of Γ_i that is intersected by the trajectory of v_i. Algorithm LinearOrderDraw either replaces an arc of e with three arcs or it replaces the inter-spine segment of e with two arcs and a new inter-spine segment (see also Figure 6); in both cases any two consecutive arcs are on opposite pages.

Let e be an edge of Γ_i incident on v_i. If $e \in E_{t,l}$ or $e \in E_{b,l}$, then after moving v_i to its target position e has both endvertices on a same spine; in this case Algorithm LinearOrderDraw replaces the inter-spine segment of e and the upper sequence of e (if such a sequence exists) with an arc having both endpoints on ℓ' (see also Figure 8); if $e \in E_{t,r}$ or $e \in E_{b,r}$, then after moving v_i to its target position, edge e has its endvertices on different spines; in this case Algorithm LinearOrderDraw replaces an arc of e having both endpoints on ℓ with an inter-spine segment plus (possibly) another arc (see also Figure 8); in both cases the new shape of e respects the definition of 2-spine drawing. It follows that Γ_{i+1} is also a 2-spine drawing. □

To complete the prof of correctness of Algorithm LinearOrderDraw, we will first prove that each 2-spine drawing Γ_i computed by Step $(i-1)$ is a planar drawing (Lemma 4), and then show that Γ_n is a 3-chain monotone topological book embedding such that the linear order of the vertices along the spine respects the given linear order (Lemma 5). The next properties are used to prove the planarity of Γ_i. We use the same notation and terminology as in the previous section.

Property 1 *The distance between ℓ and ℓ' and the interval I on ℓ' are such that: (i) the trajectory of any vertex intersects an arc with endpoints p and q only if one of the endpoints of the trajectory is in the closed interval defined by p and q; (ii) no two arcs such that one has its endpoints in the lower spine and the other has its endpoints in the upper spine can intersect.*

Proof: Let Γ_0 be the monotone topological book embedding computed at Step (-1) of Algorithm LinearOrderDraw. Let δ be the distance between the leftmost vertex and the rightmost vertex of Γ_0 along spine ℓ. The distance between ℓ and ℓ' is chosen to be greater than $\sqrt{3}\delta$ and the interval I is chosen to have length at most δ. Since an arc of an edge with endpoints p and q is drawn as a circular arc passing trough p, q, and a point of the perpendicular bisector of $\overline{pq}$ at a distance $\frac{d(p,q)}{4}$ from ℓ or ℓ', all tangent lines to each arc have slope $-\tan\frac{\pi}{6} \leq \sigma \leq \tan\frac{\pi}{6}$. By choosing the distance between ℓ and ℓ' greater than $\sqrt{3}\delta$ we have that the slope of each trajectory is either lower than $-\tan\frac{\pi}{6}$ or greater than $\tan\frac{\pi}{6}$. This implies that a trajectory intersects an arc with endpoints p and q only if one of the endpoints of the trajectory is in the closed interval defined by p and q. Also, let a be an arc of any of the drawings computed by any of the steps of Algorithm LinearOrderDraw and let p, q be the endpoints of a. By construction, p and q are inside interval I and therefore we have $d(p,q) \leq \delta$, which implies that the maximum distance between a point of a

and the spine is at most $\frac{\delta}{4}$. Since the distance between ℓ and ℓ' is larger than $\frac{\delta}{2}$ we have that no two arcs such that one has its endpoint in the lower spine and the other has its endpoints in the upper spine can intersect. $\square$

Property 2 *Let Γ_i and Γ_{i+1} be the 2-spine drawings computed by Steps $(i-1)$ and i of Algorithm* LinearOrderDraw*, respectively ($0 \leq i \leq n-1$). Every arc in the bottom page of the upper spine of Γ_{i+1} is also an arc in the bottom page of the upper spine of Γ_i.*

Proof: Step i of Algorithm LinearOrderDraw ($i = 0, 1, \ldots, n-1$) can change the shape of some edges of Γ_i by creating new inter-spine segments and new arcs. These arcs can have endpoints on the lower spine (see also Figure 6) or can be arcs in the top page of the upper spine (see also Figures 7 and 8). No arcs in the bottom page of the upper spine are created at Step i. $\square$

Property 3 *Let Γ_i be the 2-spine drawing computed by Step $(i-1)$ of Algorithm* LinearOrderDraw *($0 \leq i \leq n-1$) and let $\tau(v_i)$ be the trajectory of vertex v_i processed at Step i. Every arc of Γ_i intersected by $\tau(v_i)$ is in the top page of the lower spine of Γ_i.*

Proof: Let a be an arc of Γ_i intersected by $\tau(v_i)$. Since $\tau(v_i)$ is a straight-line segment with one endpoint in the upper spine and the other endpoint in the lower spine, arc a can either be in the top page of the lower spine or in the bottom page of the upper spine. Assume that a is an arc in the bottom page of the upper spine. By Property 2, a is also an arc of Γ_0. Since, by Theorem 4, Γ_0 is a monotone topological book embedding, if a is in the bottom page of the upper spine then the leftmost endpoint of a is a vertex of the input graph G, that we denote as v_j.

Algorithm LinearOrderDraw defines the distance between the two spines ℓ and ℓ' and the target positions along ℓ' in such a way that for every vertex v with source position $s(v)$ and target position $t(v)$, the trajectory $\tau(v)$ intersects a only if $s(v)$ is in the interval between the endpoints of a. It follows that vertex v_j is left of v_i along the spine of Γ_0, that is $j < i$.

Since Algorithm LinearOrderDraw processes the vertices in the left-to-right order along the spine of Γ_0, vertex v_j is moved to its target position before Step i is executed. Also, when the leftmost endvertex of an arc belonging to the to the bottom page of the upper spine is moved to its target position, then this arc is replaced by an inter-spine segment (see also Figure 8). It follows that a cannot be an arc of Γ_i such that a is in the bottom page of the upper spine and a is intersected by $\tau(v_i)$. $\square$

Property 4 *Let Γ_{i+1} be the 2-spine drawing computed by Step i of Algorithm* LinearOrderDraw *($0 \leq i \leq n-1$). Let a be an arc of Γ_{i+1} in the top page of the lower spine. Let y and z be the endpoints of a, with y to the left of z. Point $t(v_i)$ cannot be a point to the right of y and to the left of z.*

Proof: Two cases are possible: Either a is an arc in the top page of the lower spine also in the drawing Γ_i computed by Step $(i-1)$ or a is created at Step i. In the first case, a cannot be crossed by $\tau(v_i)$ (because otherwise a would not exist in Γ_{i+1}) and thus the property immediately holds. In the second case, either $t(v_i)$ is an endpoint of a or a is a replacing sub-edge of Step i. Then, by construction the endpoints of a are either both to the left of $t(v_i)$, or both to the right of $t(v_i)$ (see also Figure 6). $\square$

Property 5 *Let Γ_{i+1} be the 2-spine drawing computed by Step i of Algorithm* LinearOrderDraw *($0 \leq i \leq n-1$). Let a be an arc of Γ_{i+1} in the bottom page of the lower spine. Let y and z be the endpoints of a, with y to the left of z. If a is a replacing sub-edge of Step i, point $t(v_i)$ is to the right of y and to the left of z.*

Proof: Since a is a replacing sub-edge of Step i, then it replaces a sub-edge a' that is crossed by $\tau(v_i)$. Both in the case when a' is an arc and in the case when a' is an inter-spine segment, the only sub-edge that replaces a' and is in the bottom page has $t(v_i)$ between its endpoint (see also Figure 6). $\square$

Property 6 *Let Γ_{i+1} be the 2-spine drawing computed by Step i of Algorithm* LINEARORDERDRAW *($0 \leq i \leq n-1$). Let a be an inter-spine segment of Γ_{i+1}. Let y and z be the endpoints of a, with $y \in \ell$ and $z \in \ell'$. If $t(v_i)$ is to the left (right) of z, then $s(v_i)$ is to the left (right) of y.*

Proof: Two cases are possible: Either a is an inter-spine segment also in the drawing Γ_i computed by Step $(i-1)$ or a is created by Step i. In the first case, a is not crossed by the trajectory $\tau(v_i)$ (because otherwise a would not exist in Γ_{i+1}) and thus the property immediately holds. In the second case, a is a replacing sub-edge. If a replaces an arc of the upper spine, then $t(v_i)$ is one of its endpoint. Otherwise a replaces an inter-spine segment a'. Depending on whether a' is a left or a right inter-spine segment, we have that at the end of Step i either $t(v_i)$ is to the left of z and $s(v_i)$ is to the left of y , or $t(v_i)$ is to the right of z and $s(v_i)$ is to the right of y (see also Figure 6). □

Property 7 *Let Γ_i and Γ_{i+1} be the 2-spine drawings computed by Steps $(i-1)$ and i of Algorithm* LINEARORDERDRAW*, respectively ($0 \leq i \leq n-1$). Let a_1 and a_2 be two arcs of Γ_i that are both intersected by $\tau(v_i)$. If Γ_i is a planar drawing, then the sub-edges that replace a_1 and a_2 in Γ_{i+1} do not cross.*

Proof: By Property 3, both a_1 and a_2 are arcs in the top page of the lower spine of Γ_i. Let y_j and z_j be the endpoints of a_j ($j = 1, 2$), with y_j to the left of z_j. Since a_1 and a_2 are both crossed by $\tau(v_i)$ and Γ_i is planar, then y_1, y_2, z_2, and z_1 appear in this left-to-right order along ℓ' in Γ_i.

Algorithm LINEARORDERDRAW replaces a_j with three arcs a'_j (in the top page of ℓ'), a''_j (in the bottom page of ℓ'), and a'''_j (in the top page of ℓ') ($j = 1, 2$) (see also Figure 6). Denote the endpoint shared by a'_j and a''_j as p_j and the endpoint shared by a''_j and a'''_j as q_j. By construction, points y_1, y_2, p_2, p_1, q_1, q_2, z_2, and z_1 appear in this left-to-right order along ℓ' which implies that the sub-edges that replace a_1 and a_2 in Γ_{i+1} do not cross each other. □

Property 8 *Let Γ_i and Γ_{i+1} be the 2-spine drawings computed by Steps $(i-1)$ and i of Algorithm* LINEARORDERDRAW*, respectively ($0 \leq i \leq n-1$). Let a_1 and a_2 be two inter-spine segments of Γ_i that are both intersected by $\tau(v_i)$. If Γ_i is a planar drawing, then the sub-edges that replace a_1 and a_2 in Γ_{i+1} do not cross.*

Proof: Let y_j and z_j be the endpoints of a_j ($j = 1, 2$), with $y_j \in \ell$ and $z_j \in \ell'$ and assume that y_1 is to the left of y_2 on ℓ. Since Γ_i is planar, then z_1 is to the left of z_2 on ℓ'. Also, a_2 and a_1 are either both left inter-spine segments or both right inter-spine segments. Assume they are both left inter-spine segments, the other case is symmetric. Algorithm LINEARORDERDRAW replaces a_j with two arcs a'_j (in the top page of ℓ') and a''_j (in the bottom page of ℓ'), and with an inter-spine segment a'''_j ($j = 1, 2$) (see also Figure 6). Denote the endpoint shared by a'_j and a''_j as p_j and the endpoint shared by a''_j and a'''_j as q_j. By construction, points y_1, y_2, p_2, p_1, q_1, and q_2 appear in this left-to-right order along ℓ'; since z_1 is to the left of z_2 then the sub-edges that replace a_1 and a_2 in Γ_{i+1} do not cross each other. □

Property 9 *Let Γ_i and Γ_{i+1} be the 2-spine drawings computed by Steps $(i-1)$ and i of Algorithm* LINEARORDERDRAW*, respectively ($0 \leq i \leq n-1$). Let a_1 be an arc of Γ_i that is intersected by $\tau(v_i)$ and let a_2 be an inter-spine segment of Γ_i that is intersected by $\tau(v_i)$. If Γ_i is a planar drawing, then the sub-edges that replace a_1 and a_2 in Γ_{i+1} do not cross.*

Proof: By Property 3, a_1 is an arc in the top page of the lower spine of Γ_i. Let y_1 and z_1 be the endpoints of a_1, with y_1 to the left of z_1. Let y_2 and z_2 be the endpoints of a_2, with $y_2 \in \ell$ and $z_2 \in \ell'$. Since Γ_i is planar, then z_2 cannot be between y_1 and z_1. Assume that z_2 is to the left of y_1, i.e. a_2 is a left inter-spine segment (because $t(v_i)$ between y_1 and z_1). The case when z_2 is to the right of z_1, i.e. a_2 is a right inter-spine segment, is analogous.

Algorithm LINEARORDERDRAW replaces a_1 with three arcs a'_1 (in the top page of ℓ'), a''_1 (in the bottom page of ℓ'), and a'''_1 (in the top page of ℓ') (see also Figure 6). Denote the endpoint shared by a'_1 and a''_1 as p_1 and the endpoint shared by a''_1 and a'''_1 as q_1. Also, Algorithm LINEARORDERDRAW replaces a_2 with two arcs a_2' (in the top page of ℓ') and a_2'' (in the bottom page of ℓ'), and with an inter-spine segment a_2''' (see also Figure 6). Denote the endpoint shared by a_2' and a_2'' as p_2 and the endpoint shared by a_2'' and a_2''' as q_2.

By construction, points z_2, y_1, p_1, p_2, q_2, q_1, and z_1 appear in this left-to-right order along ℓ' which implies that the sub-edges that replace a_1 and a_2 in Γ_{i+1} do not cross each other. □

The following two properties consider portions of edges of Γ_i consisting of two consecutive sub-edges. Let e be an edge of Γ_i that has two consecutive sub-edges a and a' such that a is an arc in the top page of the upper spine and a' is an inter-spine segment; the portion of e consisting of a and a' is an *hook* of e. The point shared by a and a' is the *mid-point* of the hook and the other endpoint of a is the *top endpoint* of the hook. For example, edge (v_i, v_{i_0}) in Figure 8 (b) has a hook with mid-point p_0 and top endpoint v_{i_0}.

Property 10 *At any step of Algorithm* LinearOrderDraw*, an edge with an inter-spine segment shares with the upper spine at most two points. Also, if it shares two points with the upper spine, then the edge has a hook whose mid-point is to the left of the top endpoint.*

Proof: Consider the drawing Γ_0 computed at Step (-1) by Algorithm LinearOrderDraw and let $e = (v_i, v_j)$ be an edge of Γ_0 with v_i to left of v_j along the spine ℓ. Edge e shares at most three points with ℓ depending on whether it crosses or does not cross the spine of Γ_0. We recall that Algorithm LinearOrderDraw: (i) processes the vertices according to their left-to-right order along the spine of Γ_0 and (ii) at each step, changes the shape only of those edges that are intersected by the trajectory of the vertex that is moved to its target position during that step. Also, by Property 1, edge e is not intersected by the trajectory of any vertex to the left of v_i. It follows that the shape of edge e is not changed until its leftmost endvertex v_i is moved to the target position $t(v_i)$ by Step i of Algorithm LinearOrderDraw. Now consider the representation of e in the 2-spine drawing Γ_{i+1} computed by Step i of Algorithm LinearOrderDraw. Different cases are possible depending on how e is represented in the initial drawing Γ_0. Refer also Figure 8 for examples.

If in Γ_0 edge e is drawn in the bottom page of the lower spine, then in Γ_{i+1} e is drawn as an inter-spine segment connecting $t(v_i)$ with v_j. If in Γ_0 edge e is an edge in the top page of ℓ, then when Step i moves v_i to its target position, a hook η is created. The mid-point of η is a point p between $s(v_i)$ (i.e. the source position of v_i in Γ_0) and the first sub-edge endpoint that is immediately to the right of $s(v_i)$; the top endpoint of η is v_j which is to the left of $s(v_i)$, and therefore to the left of p. Finally, assume that in Γ_0 edge e intersects ℓ. Let us denote with p such intersection point; by Theorem 4, Γ_0 is a monotone topological book embedding and thus p is right of v_i and left of v_j along ℓ. At Step i, v_i is moved to its target position and a hook is created whose mid-point is p and whose top endpoint is v_j which is to the left of p. It follows that at the end of Step i, edge e has an inter-spine segment and that e satisfies the property.

At each Step h, with $i < h < j$, Algorithm LinearOrderDraw can change the shape of e by possibly introducing arcs only in its lower sequence; this happens when the trajectory $\tau(v_h)$ of the current vertex v_h that is moved to the target position either intersects the inter-spine segment or the lower sequence of e. See for example Figure 6. In no case, however, either the coordinates of the intersection points between e and ℓ are changed or new intersection points between e and ℓ are introduced. Finally at Step j, the other endvertex v_j of e is moved to ℓ' and e no longer has an inter-spine segment or hook in Γ_{j+1}. For all other steps that follow Step j, edge e will no longer have an inter-spine segment because Algorithm LinearOrderDraw moves vertices from ℓ to ℓ' and never moves them in the opposite direction. □

Property 11 *Let Γ_i be the 2-spine drawing computed by Steps $(i-1)$ of Algorithm* LinearOrderDraw *($0 \le i \le n-1$). Let e be an edge that has a hook whose top endpoint is v_i. Let e' be another edge that has an inter-spine segment a in Γ_i. Let u' be the endvertex of e' that is on ℓ and let y' be the endpoint of a on ℓ. If Γ_i is planar, then either $u' = v_i$, or y' is to the right of v_i.*

Proof: Since Algorithm LinearOrderDraw process the vertices according to their left-to-right order along ℓ, at the end of Step $(i-1)$ all vertices to the left of v_i are already moved to ℓ', i.e. v_i is the leftmost vertex on ℓ in Γ_i. Therefore either $u' = v_i$ or u' is to the right of v_i. Notice that e' may or may not have a hook. If e' has a hook then u' and y' are distinct points, otherwise they coincide. If u' and y' coincide and u' is to right of v_i, then trivially y' is to the right of v_i. If u' and y' do not coincide (i.e. e' has a hook) and u' is to right of v_i, then y' must be to the right of v_i because otherwise there would be a crossing between the two arcs of the hooks of e and e'. Thus either u' coincides with v_i or y' is to the right of v_i. □

Lemma 4 *Let Γ_i and Γ_{i+1} be the 2-spine drawings computed by Steps $(i-1)$ and i of Algorithm* LinearOrderDraw*, respectively ($0 \leq i \leq n-1$). If Γ_i is planar, then Γ_{i+1} is planar.*

Proof: Suppose by contradiction that there are two sub-edges a_1 and a_2 that cross in Γ_{i+1}. The endpoints of a_j ($j = 1, 2$) are denoted as y_j and z_j; if a_j is an arc, we shall assume that y_j is to the left of z_j; if a_j is an inter-spine segment we shall assume that y_j is in the upper spine and that z_j is in the lower spine. Since the crossing between a_1 and a_2 cannot exist in Γ_i (because it is planar by hypothesis), then at least one of the two sub-edges does not exist in Γ_i and is created at Step i of Algorithm LinearOrderDraw. A sub-edge is created at Step i either because one of its endpoints is the target position $t(v_i)$ of the vertex moved from the upper to the lower spine at that step, or because it is a replacing sub-edge.

The proof is based on a case analysis that depends on whether each of a_1 and a_2 is an inter-spine segment or an arc; if it is an arc, we also distinguish between the case that it is in the bottom or in the top page of either the upper or the lower spine. By Property 1, two arcs that have endpoints on different spines do not cross. Thus, in the case analysis below we only consider those cases in which a_1 and a_2 are both arcs with endpoints on the same spine, or at least one of them is an inter-spine segment. Also, for each case we first consider the sub-case where $t(v_i)$ is one of the endpoints of a sub-edge and then the sub-case where at least one of the sub-edges is a replacing sub-edge.

- **Both a_1 and a_2 are arcs in the top page of the lower spine.** Since a_1 and a_2 cross, y_1, y_2, z_1, and z_2 appear in this left-to-right order along ℓ' (see Figure 9(a)). By Property 4, $t(v_i)$ cannot be between y_1 and z_2. If $t(v_i) = y_1$, then there exists an inter-spine segment $\overline{a}_1$ in Γ_i having z_1 as an endpoint. This means that there is a crossing in Γ_i, which is impossible because Γ_i is planar. Analogously if $t(v_i) = z_2$, then there exists an inter-spine segment $\overline{a}_2$ in Γ_i having y_2 as an endpoint. Also in this case there is a crossing in Γ_i, which is impossible because Γ_i is planar.

  Suppose now that $t(v_i)$ is to the left of y_1. If both a_1 and a_2 are replacing sub-edges, then by Properties 7, 8, and 9 they do not cross. If only a_1 is a replacing sub-edge, then there exists a sub-edge $\overline{a}_1$ in Γ_i that is crossed by $\tau(v_i)$ and that has z_1 as one of its endpoints. This implies a crossing in Γ_i. If only a_2 is a replacing sub-edge, then a_1 is an arc of Γ_i and therefore y_1 exists also in Γ_i; since Algorithm LinearOrderDraw chooses y_2 as a point between $t(v_i)$ and the first sub-edge endpoint that follows $t(v_i)$ along ℓ', then y_2 would be to the left of y_1, thus avoiding the crossing between a_1 and a_2.

  The case when $t(v_i)$ is to the right of z_2, is symmetric to the case when $t(v_i)$ is to the left of y_1.

- **Both a_1 and a_2 are arcs in the bottom page of the lower spine.** Also in this case, a crossing is possible only if y_1, y_2, z_1, and z_2 appear in this left-to-right order along ℓ' (see Figure 9(b)). Also, none of the endpoints of a_1 and a_2 can be $t(v_i)$ because otherwise a_1 and a_2 would not be in the bottom page of the lower spine. Namely, when Algorithm LinearOrderDraw moves v_i to $t(v_i)$ all sub-edges having $t(v_i)$ as an endpoint are either arcs in the top page of the lower spine or inter-spine segments. Since at least one of a_1 and a_2 must be a replacing sub-edge, $t(v_i)$ is in the interval between y_1 and z_2 by Property 5. If both a_1 and a_2 are replacing sub-edges, then by Properties 7, 8, and 9 they do not cross. If only a_1 is a replacing sub-edge, then a_2 is a sub-edge also in Γ_i and y_2 is a point of Γ_i. Since a_1 is a replacing sub-edge in the bottom page of the lower spine, $t(v_i)$ is between y_1 and z_1; furthermore since a_2 is not a replacing sub-edge, $t(v_i)$ is in the interval between y_1 and y_2. Since Algorithm LinearOrderDraw chooses z_1 as a point between $t(v_i)$ and the first sub-edge endpoint that follows $t(v_i)$ along ℓ', then z_1 would be to the left of y_2, thus avoiding the crossing. Analogously, if only a_2 is a replacing sub-edge, then $t(v_i)$ is between z_1 and z_2; this implies that z_1 exists also in Γ_i; since Algorithm LinearOrderDraw chooses y_2 as a point between the first sub-edge endpoint that precedes $t(v_i)$ and $t(v_i)$ along ℓ', then y_2 would be to the right of z_1, thus avoiding the crossing.

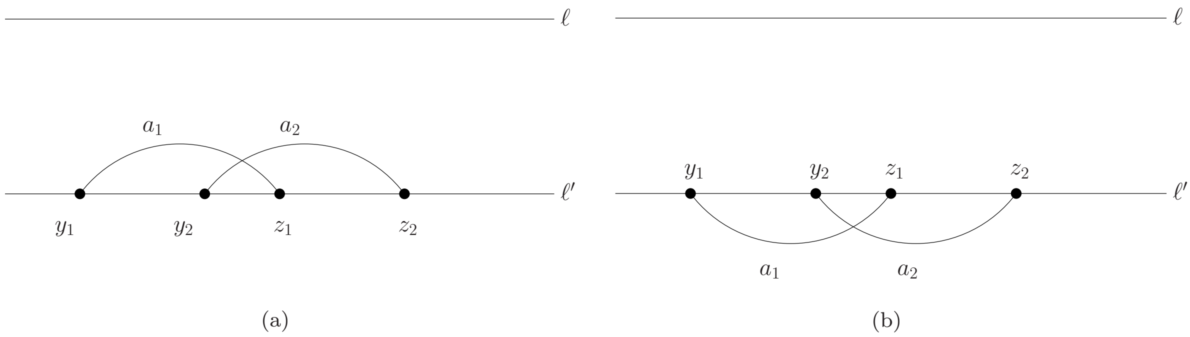


Figure 9: Two cases for the proof of Lemma 4.

- **Sub-edge a_1 is an arc in the top page of the lower spine and a_2 is an inter-spine segment.** In this case a crossing is possible only if y_1, z_2, and z_1 appear in this left-to-right order along ℓ' (see Figure 10(a)). By Property 4, $t(v_i)$ cannot be equal to z_2.

  If $t(v_i) = y_1$, then sub-edge a_1 does not exist in Γ_i and z_1 is the endpoint of an inter-spine segment $\overline{a}_1$ in Γ_i. Denote as $\overline{y}_1$ the endpoint of $\overline{a}_1$ other than z_1. On the other hand either a_2 exists in Γ_i, or there exists a inter-spine segment $\overline{a}_2$ whose endvertices are y_2 and a point $\overline{z}_2$ which is to the left of z_2. Denote as $\overline{y}_1$ the endpoint of $\overline{a}_1$ other than z_1 and let e be the edge of Γ_i that contains the inter-spine segment $\overline{a}_1$; note that $s(v_i)$ is the endvertex of e on the upper spine. Edge e may or may not have a hook. If e does not have a hook, then $s(v_i)$ and $\overline{y}_1$ coincide; otherwise $\overline{y}_1$ is to the left of $s(v_i)$ by Property 10. Also, by Property 6, $s(v_i)$ is to the left of y_2. If $s(v_i)$ and $\overline{y}_1$ coincide (i.e. e does not have a hook), then $\overline{a}_1$ crosses either a_2 or $\overline{a}_2$ in Γ_i because $s(v_i)$ is to the left of y_2 in the upper spine while z_1 is to the right of z_2 or of $\overline{z}_2$ in the lower spine. If $s(v_i)$ and $\overline{y}_1$ are distinct (i.e. e does have a hook), then $\overline{y}_1$ is to the left of $s(v_i)$ and therefore to the left of y_2; again, since z_1 is to the right of z_2 or of $\overline{z}_2$ on the lower spine, this would imply a crossing in Γ_i which is impossible.

  If $t(v_i) = z_1$, then a_1 does not exist in Γ_i and y_1 is the endpoint of an inter-spine segment $\overline{a}_1$ in Γ_i. On the other hand either a_2 exists in Γ_i, or there exists a inter-spine segment $\overline{a}_2$ whose endvertices are y_2 and a point $\overline{z}_2$ which is to the right of z_2. Denote as $\overline{z}_1$ the endpoint of $\overline{a}_1$ in Γ_i other that y_1. Point $\overline{z}_1$ is to the left of y_2 or else $\overline{a}_1$ crosses either a_2 or $\overline{a}_2$ in Γ_i, which is impossible. Let e be the edge of Γ_i that contains the inter-spine segment a_1; note that $s(v_i)$ is the endvertex of e on the upper spine. Point $s(v_i)$ must be to the right of y_2 by Property 6. Therefore e has a hook whose mid-point is $\overline{z}_1$ (to the left of y_2) and whose top endpoint is $s(v_i)$ (to the right of y_2). Let e' be the edge that contains the inter-spine segment a_2 or $\overline{a}_2$ and let u' be the endvertex of e' on ℓ. By Property 11 and based on the fact that y_2 is to the left of $s(v_i)$, it must be $u' = s(v_i)$; but in this case a_2 would not exist in Γ_{i+1}.

  Suppose now that $t(v_i)$ is to the left of y_1. If only a_1 is a replacing sub-edge, then a_2 is a sub-edge also in Γ_i and z_1 is the endpoint of a sub-edge $\overline{a}_1$ in Γ_i that can be either an arc or an inter-spine segment. Let $\overline{y}_1$ be the endpoint of $\overline{a}_1$ other than z_1. If $\overline{a}_1$ is an arc, then $\overline{y}_1$ is to the left of z_1 (otherwise $\overline{a}_1$ would not be crossed by $\tau(v_i)$), but this would imply a crossing in Γ_i, which is impossible. If $\overline{a}_1$ is an inter-spine segment, then $\overline{y}_1$ is on the upper spine. Since $\overline{a}_1$ cannot cross a_2 in Γ_i then $\overline{y}_1$ must be to the right of y_2; also, since $\tau(v_i)$ crosses $\overline{a}_1$, then $s(v_i)$ is to the right of $\overline{y}_1$ and hence to the right of y_2. However, by Property 6, $s(v_i)$ must be to the left of y_2.

  If only a_2 is a replacing sub-edge, then a_1 is a sub-edge also in Γ_i and y_1 is a point of Γ_i; since Algorithm LinearOrderDraw chooses z_2 as a point between $t(v_i)$ and the first sub-edge endpoint that follows $t(v_i)$ along ℓ', then z_2 would be to the left of y_1, thus avoiding the crossing. If both a_1 and a_2 are replacing sub-edges, then they do not cross by Properties 7, 8, and 9.

The case when $t(v_i)$ is to the right of z_1, is symmetric to the case when $t(v_i)$ is to the left of y_1.

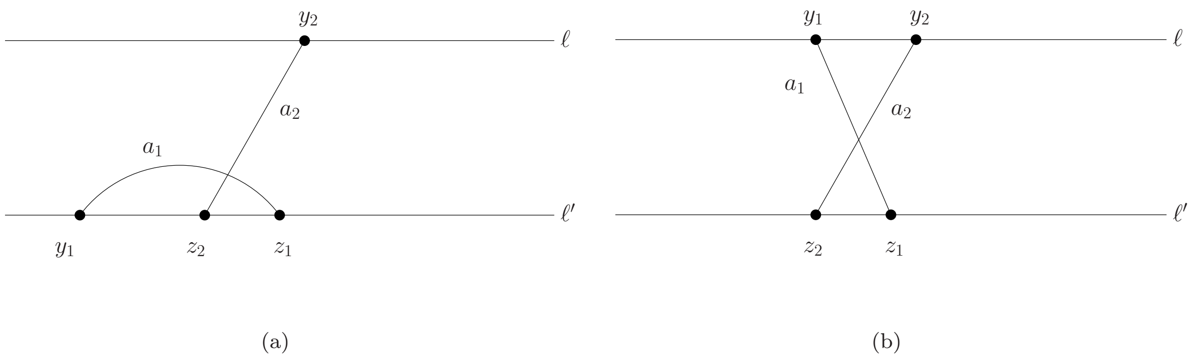


Figure 10: Two cases for the proof of Lemma 4.

- **Both a_1 and a_2 are inter-spine segments.** In this case, a crossing is possible only if y_2 is to the right of y_1 (on ℓ) and z_2 is to the left of z_1 (on ℓ') (see Figure 10(b)).

  If $t(v_i) = z_1$, then a_1 does not exist in Γ_i and the endvertices of the edge e that contains a_1 are both in the upper spine. Also, one of the endvertices of e is v_i. By Property 6 $s(v_i)$ is to the right of y_2. and therefore to the right of y_1. Notice that in no case Algorithm LinearOrderDraw moves the first endvertex of an edge to ℓ' creating an inter-spine segment with an endpoint on ℓ that is to the left of the moved point. This implies that this case never happens.

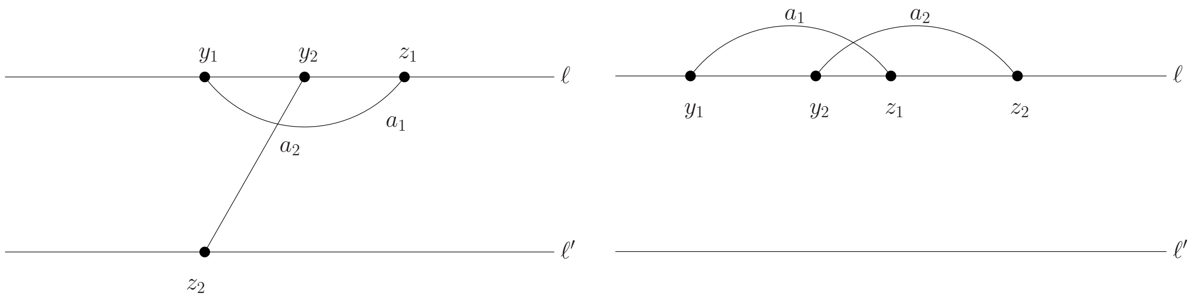
  If $t(v_i) = z_2$, then a_2 does not exist in Γ_i and the endvertices of the edge e that contains a_2 are both in the upper spine. Also, one of the two endvertices is v_i. By Property 6 $s(v_i)$ is to the left of y_1. Denote by u the endvertex of e other than v_i. By Property 10, either $s(u) = y_2$ or y_2 is to the left of $s(u)$ in Γ_{i+1}. If $s(u) = y_2$ in Γ_{i+1}, then e is represented in Γ_i by an arc a in the bottom page of the upper spine; the endpoints of a are y_2 (to the right of y_1) and $s(v_i)$ (to the left of y_1). But this means that there is a crossing in Γ_i between a and a_1, which is impossible. If y_2 is to the left of $s(u)$ in Γ_{i+1}, then e is represented in Γ_i by two arcs: a' in the bottom page of the upper spine and a'' in the top page of the upper spine; the endpoints of a' are $s(v_i)$ (to the left of y_1) and y_2 (to the right of y_1), while the endpoints of a'' are y_2 and $s(u)$ (both to the right of y_1). In this case there would be a crossing in Γ_i between a_1 and a', which is impossible.

  If $t(v_i)$ is between z_1 and z_2 then, by Property 6, $s(v_i)$ should be to the left of y_1 and to the right of y_2, which is impossible.

  Suppose now that $t(v_i)$ is to the left of z_2. If only a_1 is a replacing sub-edge, then a_2 is a sub-edge also in Γ_i and z_2 is a point of Γ_i; since Algorithm LinearOrderDraw chooses z_1 as a point between $t(v_i)$ and the first sub-edge endpoint that follows $t(v_i)$ along ℓ', then z_1 would be to the left of z_2, thus avoiding the crossing. If only a_2 is a replacing sub-edge, then a_1 is a sub-edge also in Γ_i and y_2 is the endpoint of an inter-spine segment $\overline{a}_2$ in Γ_i. Let $\overline{z}_2$ be the endpoint of $\overline{a}_2$ other than y_2. Since $\tau(v_i)$ crosses $\overline{a}_2$, then $\overline{z}_2$ is to the left of z_1. But in this case a_1 and $\overline{a}_2$ would cross in Γ_i, which is impossible. If both a_1 and a_2 are replacing sub-edges, then they do not cross by Properties 7, 8, and 9.

  The case when $t(v_i)$ is to the right of z_1 is symmetric to the case when $t(v_i)$ is to the left of z_2.

- **Sub-edge a_1 is an arc in the bottom page of the upper spine and a_2 is an inter-spine segment.** In this case a crossing is possible only if y_2 is between y_1 and z_1 (see Figure 11(a)). By Property 2 a_1 exists also in Γ_i. Thus a_2 must not exist in Γ_i (otherwise the crossing between a_1 and a_2 would exist in Γ_i). If $t(v_i) \neq z_2$, then a_2 is a replacing sub-edge and y_2 is the endpoint of an inter-spine segment $\overline{a}_2$ of Γ_i. Since y_2 is between y_1 and z_1 there would be a crossing between $\overline{a}_2$ and a_1 in Γ_i. Hence $t(v_i) = z_2$. Note that, since a_2 is an arc in the bottom page of the upper spine, y_1 is not a spine crossing but a "real" vertex v_j. This means that $s(v_i)$ is to the left of y_1 because otherwise Algorithm LinearOrderDraw would have moved $v_j = y_1$ to ℓ' at some Step j, with $j < i$ and arc a_1 would not exist in Γ_{i+1}. Let e be the edge that contains a_2, let u be the endvertex of e that is on ℓ. Edge e may or may not have a hook. If e does not have a hook, then u and y_2 coincide; otherwise y_2 is to the left of u in Γ_{i+1} by Property 10. If u and y_2 coincide, i.e. if e does not have a hook, then e is represented in Γ_i by an arc a in the bottom page of the upper spine; the endpoints of a are y_2 (between y_1 and z_1) and $s(v_i)$ (to the left of y_1). But this implies that there is a crossing in Γ_i between a and a_1, which is impossible. If y_2 is to the left of $s(u)$, i.e. e does have a hook, then e is represented in Γ_i by two arcs: a' in the bottom page of the upper spine and a'' in the top page of the upper spine; the endpoints of a' are $s(v_i)$ (to the left of y_1) and y_2 (between y_1 and z_1), while the endpoints of a'' are y_2 and $s(u)$ (to the right of z_1). In this case there would be a crossing in Γ_i between a_1 and a', which is impossible.



(a) (b)

Figure 11: Two cases for the proof of Lemma 4.

- **Both a_1 and a_2 are arcs in the bottom page of the upper spine.** By Property 2 both a_1 and a_2 are arcs of Γ_i, but this would imply that the crossing exists in Γ_i, which is impossible.

- **Both a_1 and a_2 are arcs in the top page of the upper spine.** In this case, a crossing is possible only if y_1, y_2, z_1, and z_2 appear in this left-to-right order along ℓ (see Figure 11(b)). At least one of the two sub-edges a_1 and a_2 must not exist in Γ_i (otherwise there would be a crossing in Γ_i). Let e_1 and e_2 be the edges that contain sub-edge a_1 and a_2, respectively.

  Suppose first that a_1 does not exist in Γ_i, while a_2 does. In this case e_1 has both endvertices on ℓ in Γ_i while it has one endvertex on ℓ' in Γ_{i+1}, i.e. v_i is the leftmost endvertex of e_1. Since a_1 is an arc in the top page of the upper spine, then e_1 is drawn in Γ_i either as an arc a in the top page of ℓ, or as two arcs a' and a'' such that a' is in the bottom page of ℓ and a'' is in the top page of ℓ. In both cases there exists an arc in Γ_i that is in the top page of ℓ, that has one endpoint to the left of y_2 and that has the second endpoint between y_2 and z_2. This implies that there is a crossing in Γ_i, which is impossible.

  Suppose now that a_2 does not exist in Γ_i, while a_1 does. In this case e_2 has both endvertices on ℓ in Γ_i while it has one endvertex, which is v_i, on ℓ' in Γ_{i+1}, i.e. v_i is the leftmost endvertex of e_1. Since a_2 is an arc in the top page of the upper spine, then e_2 is drawn in Γ_i either as an arc a in the top page of ℓ, or as two arcs a' and a'' such that a' is in the bottom page of ℓ and a'' is in the top page of ℓ. In the first case the leftmost endpoint of a is $s(v_i)$ and, due to the fact that there is no crossing in Γ_i, $s(v_i)$ must be to the left of y_1. Since Algorithm LinearOrderDraw chooses y_2 as a point between $s(v_i)$ and the first sub-edge endpoint that follows $s(v_i)$ along ℓ, then y_2 would be to the left of y_1, thus avoiding the crossing. In the second case the leftmost endpoint of a' is $s(v_i)$ and, due to the fact that there is no crossing in Γ_i, the rightmost endpoint of a', call it z', must be to the left of y_1. The endpoints of a'' are z' (to the left of y_1) and z_2 (to the right of z_1). When Algorithm LinearOrderDraw moves v_i to $t(v_i)$, arc a' is replaced by an inter-spine segment while arc a'' is unchanged and coincides with a_2; but this would imply that a_2 and a_1 do not cross.

Finally, suppose that both a_1 and a_2 do not exist in Γ_i. In this case both e_1 and e_2 have both endvertices on ℓ in Γ_i while they have one endvertex, which is v_i, on ℓ' in Γ_{i+1}, i.e. v_i is the leftmost endvertex of both e_1 and e_2. Since a_j is an arc in the top page of the upper spine, then e_j is drawn in Γ_i either as an arc a'_j in the top page of ℓ, or as two arcs a'_j and a''_j such that a'_j is in the bottom page of ℓ and a''_j is in the top page of ℓ ($j = 1, 2$). Assume first that both e_1 and e_2 are drawn in Γ_i as arcs in the top page. The endvertex of e_1 other that v_i is z_1, and the endvertex of e_2 other than v_i is z_2. Then, by construction, the endpoint y_1 of a_1 is chosen to be to the left of the endpoint y_2 of a_2, thus avoiding the crossing. Assume now that both e_1 and e_2 are drawn in Γ_i as two arcs. In this case a_1 coincides with a''_1 and a_2 coincides with a''_2. Therefore a crossing between a_1 and a_2 would imply a crossing between a''_1 and a''_2, which is impossible because Γ_i is planar. Finally, assume that e_1 is drawn in Γ_i as an arc in the top page and that e_2 is drawn in Γ_i as two arcs (the case when e_1 is drawn as two arcs and e_2 as an arc in the top page is analogous). In this case, since Γ_i is planar, either the endpoints of a''_2 are both to the right of both the endpoints of a'_1, or they are both between the endpoints of a'_1. Arc a_2 coincides with a''_2, while the rightmost endpoint of a_1 coincides with the rightmost endpoint of a'_1 and the leftmost point of a_1 is a point between $s(v_i)$ and the leftmost point of a'_1. It follows that either the endpoints of a_2 are both to the right of both the endpoints of a_1, or they are both between the endpoints of a_1. In both cases a crossing between a_1 and a_2 is not possible.

From the case analysis above it follows that the assumption that two sub-edges cross in Γ_{i+1} always leads to a contradiction, which implies that the statement is true. □

Lemma 5 *Let G be a planar graph with n vertices and let λ be a linear ordering of the vertices of G. Algorithm* LinearOrderDraw *computes a 3-chain topological book embedding Γ' of G such that the left-to-right order of the vertices in Γ' coincides with λ.*

Proof: By Lemmas 3 and 4 the output $\Gamma' = \Gamma_n$ of Algorithm LinearOrderDraw is a planar 2-spine drawing. Also, all vertices of G are on the spine ℓ' of Γ'. Thus Γ' is a topological book embedding. The left-to-right order of the vertices of G in Γ' is equal to λ by construction because Algorithm LinearOrderDraw defines the target positions of the vertices according to λ. It remains to show that every edge of Γ' consists of at most three x-monotone chains.

Let $e = (v_i, v_j)$ be an edge of Γ_0 such that v_i is to the left of v_j along the upper spine ℓ of Γ_0. By Property 1, edge e is not intersected by the trajectory of any vertex to the left of v_i. It follows that for any Step h of Algorithm LinearOrderDraw such that $0 \leq h \leq i-1$ e is an edge with both endvertices on the upper spine ℓ and the shape of e is not changed. Also, for any Step h of Algorithm LinearOrderDraw such that $j+1 \leq h \leq n-1$ both endvertices of e are on the lower spine ℓ', e does not have any inter-spine segment, and the shape of e is changed if the trajectory $\tau(v_h)$ crosses some of the arcs of e. Let a be an arc of e crossed by $\tau(v_h)$. By Property 3 a is in the top page of the lower spine. As also illustrated in Figure 6, Algorithm LinearOrderDraw replaces a with an x-monotone portion consisting of three arcs such that the leftmost endpoint and the rightmost endpoint of this portion coincide with the leftmost endpoint and rightmost endpoint of a, respectively. It follows that the number of x-monotone chains that form edge e is not changed by any Step h such that $j+1 \leq h \leq n-1$. We now prove that the number of x-monotone chains of e created from Step i to Step j is at most three. There are three cases to consider, depending on the drawing of e in Γ_0.

If e is drawn in Γ_0 as an arc in the bottom page of the spine ℓ, then at Step i it is transformed into an inter-spine segment and the lower sequence of e is empty (i.e. there are no arcs in the lower sequence); see also Figures 12(a) and 12(b). New arcs are added to the lower sequence of e by Step h of Algorithm LinearOrderDraw with $i+1 \leq h \leq j-1$ only if $\tau(v_h)$ crosses the inter-spine segment or an arc of the lower

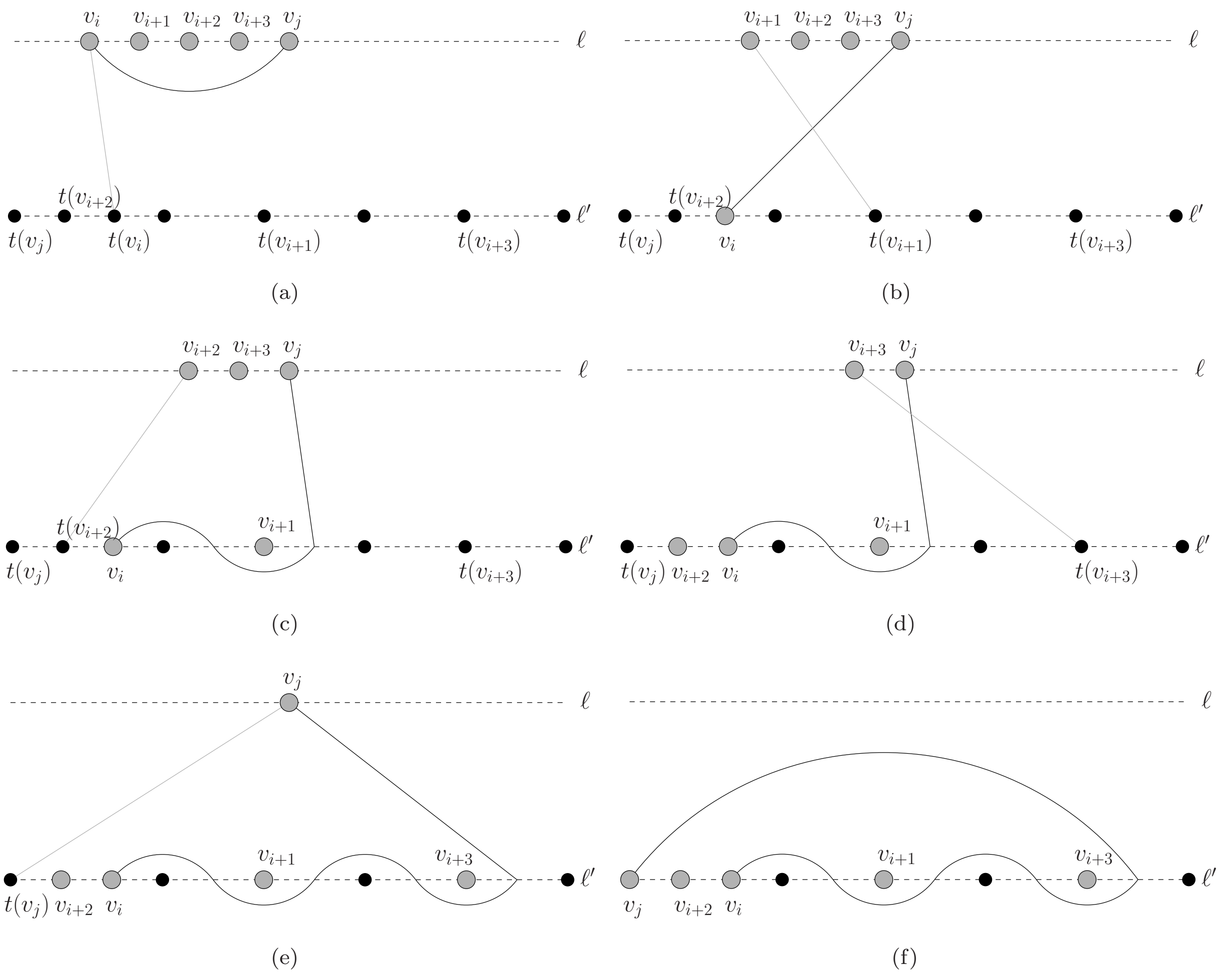


Figure 12: (a) An edge e in the bottom page of ℓ in Γ_0. (b) The edge after that Step i of Algorithm LinearOrder-Draw is executed and a trajectory crossing it; the inter-spine segment is a left inter-spine segment. (c)–(e) New arcs are added to the lower sequence of e. (f) At the end Step j, e consists of at most two x-monotone chains.

sequence of e. By the same reasoning as above, when an arc is crossed by $\tau(v_h)$ the number of x-monotone chains of the lower sequence of e does not change. So, assume that $\tau(v_h)$ crosses the inter-spine segment of e and refer to Figures 12(b), 12(c), 12(d), and 12(e). Note that since e is in the lower page of the spine ℓ of Γ_0, the inter-spine segment crossed by $\tau(v_h)$ is a left inter-spine segment. Algorithm LinearOrderDraw modifies the lower sequence of e by concatenating it with an x-monotone portion consisting of two arcs and such that the leftmost endpoint of this x-monotone portion coincides with the rightmost endpoint of the lower sequence. It follows that the end of Step $j-1$, the lower sequence of e is an x-monotone chain. Let p be the rightmost endpoint of such x-monotone chain; see also Figures 12(e) and 12(f). At Step j, vertex v_j is moved to its target position $t(v_j)$. If $t(v_j)$ is to the right of p, then another arc is added to the lower sequence such that the added arc shares its leftmost endpoint with the lower sequence and thus e consists of one x-monotone chain. If otherwise $t(v_i)$ is not to the left of p, the new arc added at Step j and the lower sequence of Step $j-1$ form two x-monotone chains. Therefore, if e is drawn in Γ_0 as an edge in the bottom page of ℓ, then e consists of at most two x-monotone chains in Γ'.

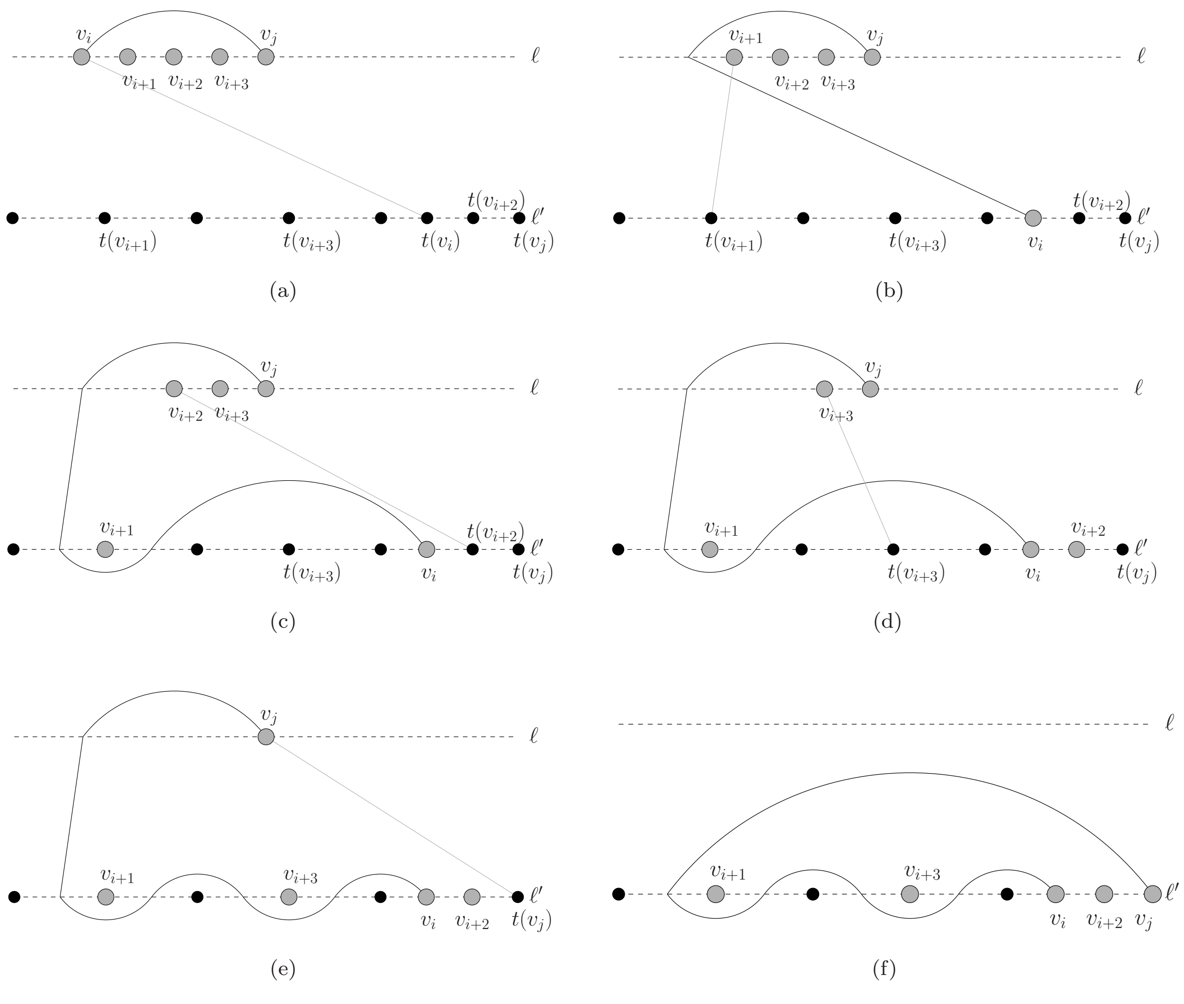


Figure 13: (a) An edge e in the top page of ℓ in Γ_0. (b) The edge after that Step i of Algorithm LinearOrderDraw is executed and a trajectory crossing it; the inter-spine segment is a right inter-spine segment. (c)–(e) New arcs are added to the lower sequence of e. (f) At the end Step j, e consists of at most two x-monotone chains.

If e is drawn in Γ_0 as an arc in the top page of the spine ℓ, then at Step i it is transformed into an hook and the lower sequence of e is empty; see also Figures 13(a) and 13(b). The reasoning is symmetric to the one of the previous case, the main difference being that in this case the inter-spine segment crossed by $\tau(v_h)$ is a right inter-spine segment and that the lower sequence is increased by adding two arcs at its leftmost end. See also Figures 13(b), 13(c), 13(d), and 13(e). Therefore, if e is drawn in Γ_0 as an edge in the top page of ℓ, then e consists of at most two x-monotone chains in Γ'.

Finally, assume that e is drawn in Γ_0 as two arcs, on in the bottom page and the other in the top page of the spine ℓ. By adding a dummy vertex at the spine crossing between e and ℓ, the edge e can be regarded as the concatenation of two edges, one in the bottom page of ℓ and the other in the top page of ℓ. As illustrated in Figure 14, by combining the previous arguments, it is straightforward to conclude that at the end of Step j edge e consists of at most three x-monotone chains. Therefore, if e is drawn in Γ_0 as an edge that crosses the spine, then e consists of at most three x-monotone chains in Γ'. $\square$

Theorem 5 *Let G be a planar graph with n vertices and let λ be a linear ordering of the vertices of G. There exists an $O(n^2 \log n)$-time algorithm that computes a 3-chain topological book embedding Γ' of G such that the left-to-right order of the vertices in Γ' coincides with λ.*

Proof: By Lemma 5, the drawing Γ' computed by Algorithm LinearOrderDraw is a 3-chain topological book embedding Γ' of G such that the left-to-right order of the vertices along the spine of Γ' is λ.

As for the time complexity, computing a monotone topological book embedding of the input graph can be done in $O(n)$ time by Theorem 4; also computing the target positions of the vertices along ℓ' can be done in $O(n)$ time.

At Step i ($0 \leq i \leq n-1$), Algorithm LinearOrderDraw performs the following tasks: (1) It finds the sub-edges crossed by the trajectory $\tau(v_i)$. (2) It modifies the shape of these sub-edges. (3) It modifies the shape of the edges incident to v_i.

We can use an AVL-tree for each of the x-monotone chains of each edge. Since each edge has at most three x-monotone chains and G is planar, we have $O(n)$ such AVL-trees. Let T be an AVL-tree associated with an x-monotone chain π of an edge. Each node of T stores an arc a of π; a is represented by the x-coordinates of its endpoints and by a flag that describes whether a is in the top or in the bottom page of ℓ'. Since at each step of Algorithm LinearOrderDraw a constant number of arcs can be added to an x-monotone chain and the total number of steps is $O(n)$, the number of nodes of T is $O(n)$.

Task (1) can be executed as follows. By performing a search operation on T, one can find in $O(\log n)$ time the arc of π that is intersected by $\tau(v_i)$, if such an arc exists. Also, deciding if an inter-spine segment is crossed by $\tau(v_i)$ can be done in $O(1)$ time by comparing the coordinates of the endpoints of the inter-spine segment with the coordinates of $s(v_i)$ and $t(v_i)$. Therefore, finding all the sub-edges of an edge crossed by $\tau(v_i)$ can be done in $O(\log n)$ time. Since we have $O(n)$ edges, the overall time complexity of Task (1) is $O(n \log n)$.

Task (2) modifies the shape of the edges found in Task (1). For each edge, the sub-edges intersected by $\tau(v_i)$ are replaced by a constant number of sub-edges. This implies computing the endpoints of the new sub-edges, which can be done in $O(1)$ time for each sub-edge, and to update the AVL-trees associated with the x-monotone chains that are changed. Since each x-monotone chain can be crossed by $\tau(v_i)$ at most once and each crossing requires a constant number of updates in the associated AVL-tree, Task (2) can be executed in $O(\log n)$ time per edge, i.e. in $O(n \log n)$ time in total.

Also Task (3) requires to compute the endpoints of new sub-edges, which can be done in $O(1)$ time for each sub-edge and it may require to update an AVL-tree per edge (this happens when the inter-spine segment and the upper sequence of an edge are replaced by an arc in the lower sequence). Thus, also Task (3) can be executed in $O(n \log n)$ time. It follows that Step i can be executed in $O(n \log n)$. Since there are n such steps, Algorithm LinearOrderDraw can be executed in $O(n^2 \log n)$ time. $\square$

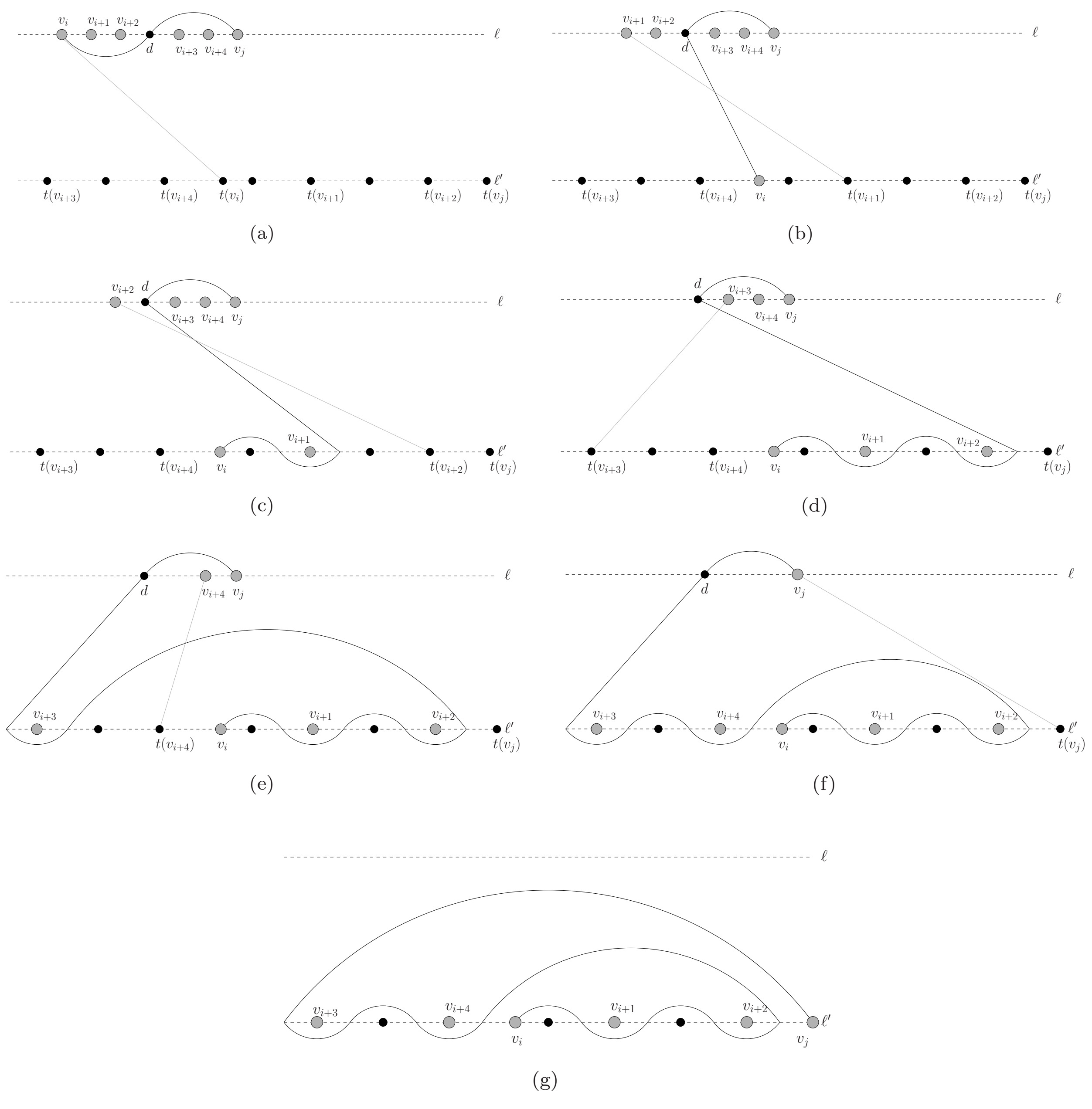


Figure 14: (a) An edge e that crosses spine of Γ_0. (b) The edge after Step i Algorithm LinearOrderDraw. (c)–(e) New arcs are added to the lower sequence of e. (f) At the end Step j, e consists of at most three x-monotone chains.

7 Upper Bounds on the Curve Complexity of k-colored Point-set Embeddings

Let G be a k-colored planar graph with n vertices such that $2 \leq k \leq n$ and let S be a k-colored set of points compatible with G. In this section we first show how to compute a k-colored point-set embedding of G on S having curve complexity at most $3n + 2$ and then consider the special case in which there are $k - 1$ colors each associated with exactly one vertex in G and the remaining color is associated with all remaining vertices of G.

7.1 Computing k-colored Point-set Embeddings

Let σ be the k-colored sequence induced by S. Based on Theorem 3, we prove an upper bound on the curve complexity of a k-colored point-set embedding of G on S by computing an external augmenting k-colored Hamiltonian path $\mathcal{H}''$ of G consistent with σ and inducing a bounded number of division vertices per edge. To this aim, we consider the linear order given by σ and exploit Algorithm LinearOrderDraw to compute $\mathcal{H}''$. As it will explained later, the number of division vertices per edge depends on the number of spine crossings. We start with two lemmas that show how to simplify the shape of the edges computed by Algorithm LinearOrderDraw in order to reduce this number of spine crossings.

The *interval* of an arc of a topological book embedding is the open interval of the spine between the leftmost endpoint and the rightmost endpoint of the arc.

Lemma 6 *Let G be a planar graph with n vertices and let Γ' be the 3-chain topological book embedding computed by Algorithm* LinearOrderDraw*. Γ' can be transformed into a new 3-chain topological book embedding Γ'' of G such that: (i) the left-to-right order of the vertices along the spine is the same as in Γ' and (ii) every x-monotone chain crosses the spine at most $n - 1$ times.*

Proof: Assume first that for every arc of Γ' the interval of the arc contains at least a vertex of G. Since there are n vertices, every x-monotone chain π of every edge of Γ' consists of at most n arcs. Since a spine crossing of an x-monotone chain is defined by two consecutive arcs, π can cross the spine at most $n-1$ times and the proof is completed by saying that Γ'' coincides with Γ'. So, assume otherwise that there is at least one arc a of Γ' such that the interval of a does not contain a vertex of G; we show how to modify the shape of some of the edges of Γ' in order to construct Γ''.

We first observe that arc a belongs to the top page of Γ'. Namely, by construction, every arc that is drawn in the bottom page of ℓ' by Algorithm LinearOrderDraw is such that its interval contains at least one vertex of G (see also Section 6.1). This implies that the interval of a cannot contain both endpoints of arcs in the bottom page of Γ' or else the interval of a would also contain a vertex of G. If the interval of a contains one endpoint u of an arc a' of the bottom page, then it must contain both endpoints of the arc a'' that shares the endpoint u with a'. Indeed, since Γ' is a topological book embedding, two arcs that share an endpoint are in opposite pages; hence, a'' is in the top page of Γ' and, by the planarity of Γ', it must have both endpoints in the interval of a. Also, a'' cannot have any vertex of G in its interval, or else also a would have a vertex in its interval. It follows that either the interval of a does not contain any endpoint of any arc of Γ' or there exist arcs of the top page that have both endpoints in the interval of a and that do not contain any vertex in their interval.

By iterating this argument we conclude that if there is at least one arc of Γ' whose interval does not contain a vertex of G, it must exists at least one arc of the top page of Γ' such that its interval does not contain any endpoints. We say that such an arc is *good for simplification*. Let π be an x-monotone chain of an edge e of Γ' such that π consists of at least two arcs (i.e. π crosses the spine). Assume that π contains at least one arc good for simplification and let a be one of such arcs. Let y be the left endpoint of a and z be the right endpoint of a. Note that since π consists of at least two arcs, at least one of the endpoints of a cannot be an endpoint of π.

We consider first the case that neither y nor z are endpoints of π. Let a_1 be the arc immediately preceding a along π, i.e. let a_1 be the arc of π whose rightmost endpoint is y; denote as y_1 the other endpoint of a_1.

Similarly, let z and z_2 be the endpoints of the arc a_2 that immediately follows a along π. Recall that by definition of 3-chain topological book embedding consecutive arcs are in opposite pages and hence a_1 and a_2 are in the bottom page. Chain π is modified by deleting arcs a_1, a, and a_2 and by inserting a single arc a' such that the endpoints of a' are y_1 and z_2 and a' is in the bottom page. This procedure is illustrated in Figures 15(a) and 15(b). Note also that a' is in the page opposite to that of the arc preceding a_1 and of the arc following a_2 in Γ'.

If only y (only z) is an endpoint of π, let a_1 be the arc of π that follows (precedes) a, i.e. let a_1 be the arc of π whose leftmost (rightmost) endpoint is y; denote as z_1 (y_1) the other endpoint of a_1. Chain π is modified by deleting arcs a_1 and a and by inserting a single arc a' such that the endpoints of a' are y and z_1 (y_1 and z) and a' is in the bottom page. This procedure is illustrated in Figures 15(c), 15(d), 15(e), and 15(f). Also in this case, arc a' is in the opposite page of the arc following (preceding) a_1 in Γ'.

Let $\hat{\Gamma}$ be the drawing after a has been removed by the simplification procedure described above. $\hat{\Gamma}$ differs from Γ' only for the shape of edge e. Let π' be the x-monotone portion of Γ' that is replaced in $\hat{\Gamma}$ by arc a' (π' is formed by arc a and by one or two of its neighboring arcs depending on the cases described above). Since the endpoints of a' coincide with the endpoints of π' and since an arc is clearly an x-monotone curve, it follows that the number of x-monotone chains that form e is the same in Γ' and in $\hat{\Gamma}$.

We now show that replacing π' with arc a' does not introduce edge crossings. Consider first the case that π' is formed by three arcs, i.e. neither y nor z are endpoints of π. Arcs a_1, a, and a_2 are replaced by a single arc a' in the bottom page. We show that the replacement does not create any crossing. If a' crossed another arc, this arc should be in the bottom page. Since a is good for simplification, no arcs have endpoints in the interval of a. Also, by the planarity of Γ', we have that an arc in the bottom page of Γ' has both endpoints either to the left of y_1, or to the right of z_2, or it has one endpoint to the left of y_1 and the other one to the right of z_2. It follows that arc a' does not cross any arc in its page. The proof that $\hat{\Gamma}$ is a planar drawing in the case that π' consists of two arcs is analogous.

From the arguments above it follows that $\hat{\Gamma}$ is a 3-chain topological book embedding of G; also, since the coordinates of the vertices of G have not been changed when transforming Γ' into $\hat{\Gamma}$, the left-to-right order of the vertices along the spine is the same in the two drawings. We now look for arcs good for simplification in $\hat{\Gamma}$. If there are no such arcs, we say that the wanted Γ'' coincides with $\hat{\Gamma}$. If otherwise $\hat{\Gamma}$ has an arc good for simplification, we apply the above described simplification procedure to this arc and obtain a new 3-chain topological book embedding of G that maintains the left-to-right order of the vertices along the spine as in $\hat{\Gamma}$ and hence as in Γ'. The procedure is then repeated until a 3-chain topological book embedding of G, that we call Γ'', is computed such that Γ'' does not have an arc good for simplification and it maintains the same left-to-right order of the vertices along the spine as in Γ'. Observe that the interval of every arc of Γ'' contains at least one vertex of G because otherwise, by the argument at the beginning of this proof, there would be at least one arc good for simplification in Γ''. It follows that every x-monotone chain of Γ'' crosses the spine at most $n-1$ times and that Γ'' satisfies the statement. □

The procedure described in the proof of Lemma 6 will be called *simplification procedure* in the remainder. Also we call the drawing computed by the simplification procedure a *simplified* 3*-chain topological book embedding* of G. The next lemma discusses the time complexity of the simplification procedure and will be used to prove an upper bound on the time complexity of computing k-colored point-set embeddings.

Lemma 7 *Let G be a planar graph with n vertices and let λ be a linear ordering of the vertices of G. There exists an $O(n^2 \log n)$-time algorithm that computes a simplified* 3*-chain topological book embedding Γ'' of G such that the left-to-right order of the vertices in Γ'' coincides with λ.*

Proof: We use the same definitions and notation as in the proof of Lemma 6. By Theorem 5, one can compute in $O(n^2 \log n)$ time a 3-chain topological book embedding Γ' of G where the left-to-right order of the vertices along the spine is λ. By using Lemma 6, we can compute a simplified 3-chain topological book embedding Γ'' of G such that the left-to-right order of the vertices in Γ'' is the same as in Γ'. As explained in the proof of Lemma 6, after an arc of Γ' that is good for simplification is processed a new drawing if G is constructed that can have some other arc good for simplification. We say that an arc is *candidate for simplification* if it is good for simplification or if it will become good for simplification at some step of

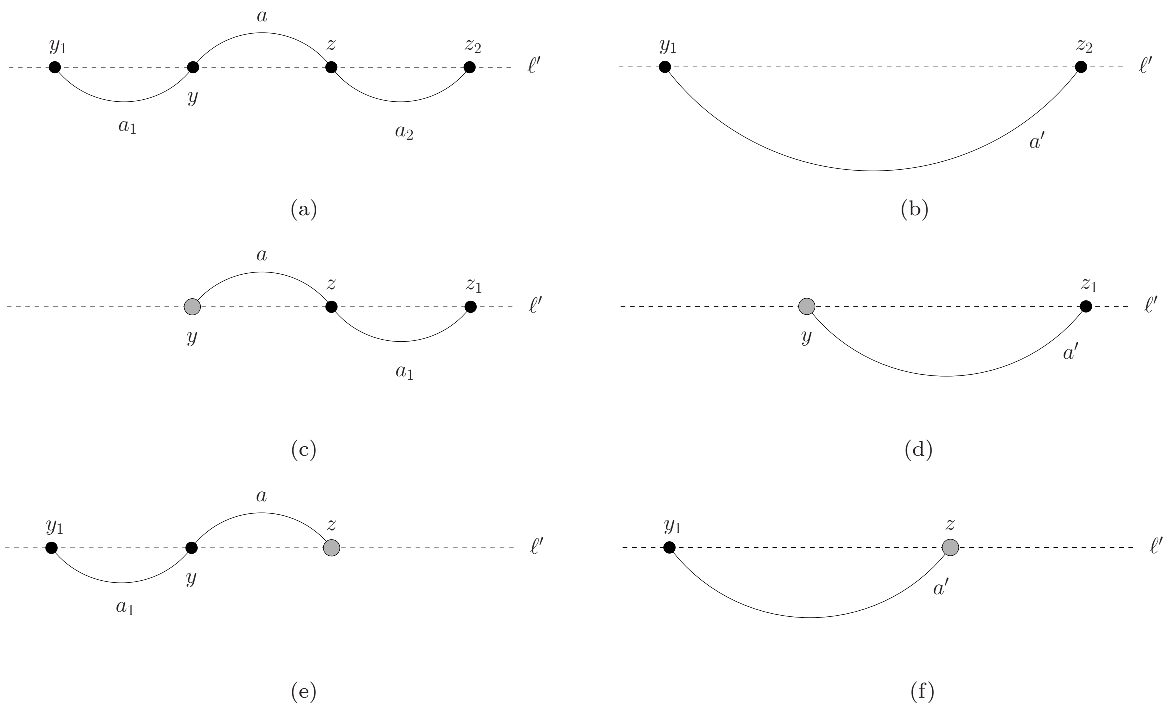


Figure 15: Illustration of the simplification procedure described in the proof of Lemma 5.

the simplification procedure. The proof of the upper bound on the time complexity of the simplification procedure relies on a characterization of those arcs that are candidate for simplification.

Firstly, observe that if an arc is candidate for simplification then it belongs to an x-monotone chain of Γ' having at least two arcs; indeed, by definition, an arc good for simplification belongs to an x-monotone chain of at least two arcs. Secondly, as already discussed in the proof of Lemma 6, no arc in the bottom page of Γ' is good for simplification. Furthermore, since any step of the simplification procedure neither changes the coordinates of the vertices in the drawing nor it changes the coordinates of those spine crossings that are not removed, it follows that arc in the bottom page of Γ' will never be good for simplification at any of the steps of the simplification procedure. By the same reasoning, an arc in the top page of the upper page of Γ' whose interval contains some vertex of G will never be good for simplification.

Thirdly, let a be an arc in the top page of Γ' such that a is good for simplification and let a' be another arc in the top page of Γ' such that the interval of a' contains only the endpoints of a. After the simplification procedure is applied to arc a, we have that a' is still an arc in the resulting drawing (the simplification procedure only deletes a and one or two arcs adjacent to a and belonging to the bottom page). Also, the interval of a' does not contain any other endpoint of any other arcs, because this interval only contained the endpoints of a and a has been replaced by an arc in the bottom page whose endpoints are out of the interval of a'. By iterating this reasoning, we can conclude that if an arc of the top page of Γ' is such that its interval only contains endpoints that correspond to spine crossings, then this arc will become good for simplification and thus it is candidate for simplification.

From the observations above it follows that an arc is candidate for simplification if and only if it satisfies the following three conditions: (i) it belongs to an x-monotone chain of Γ' having at least two arcs, (ii) it belongs to the upper page of Γ', and (iii) its interval does not contains any vertex of G. By using this characterization, the simplification algorithm can be implemented as follows.

As described in the proof of Theorem 5, we can assume to have an AVL tree for each x-monotone chain of Γ' such that each node of the AVL tree stores an arc of the chain. We construct two arrays, one storing all vertices of Γ' and the other one storing all endpoints (vertices and spine crossings); both arrays are sorted according to the left-to-right order of their elements along the spine of Γ'. We call A the array with only the vertices of Γ' and A' the array with the vertices and the spine crossings of Γ'. As observed in the proof of Theorem 5, the total number of arcs stored in the AVL trees of Γ' is $O(n^2)$ and thus the two arrays A and A' can be constructed in $O(n^2 \log n)$ time.

We now visit those arcs that are in the top page and are stored in those AVL trees having more than one element (an AVL tree having only one element corresponds to an x-monotone chain of a single arc). Let a be the currently visited arc; by performing a binary search in A, we determine in $O(\log n)$ time whether a is candidate for simplification. In the affirmative case, we equip the two elements of A' that are the leftmost endpoint and the rightmost endpoint of a with a reference to a. Since there are $O(n^2)$ arcs in the top page of Γ', it follows that identifying the arcs candidate for simplification and equipping the elements of A' with pointers to them can be done in $O(n^2 \log n)$ time.

We now visit all endpoints of Γ' from left-to-right by scanning A'. An arc a that is candidate for simplification is processed only when its rightmost endpoint is encountered. This guarantees that all other candidate arcs whose endpoints are in the interval of a have been already processed and that a is now good for simplification. Processing a consists of identifying one or two arcs that precede and follow a in Γ', deleting both a and such arcs, and replacing the (two or three) deleted arcs with a single arc in the bottom page. All these steps can be executed in $O(\log n)$ time by accessing and updating the AVL tree of a.

It follows that the overall time complexity of the simplification procedure of Lemma 6 can be done in $O(n^2 \log n)$ time. □

Lemma 8 *Let G be an n-colored planar graph with n vertices and let σ be an n-colored sequence compatible with G. G admits an external augmenting n-colored Hamiltonian path consistent with σ inducing at most $3n-3$ flat division vertices and at most 2 pointy division vertices per edge.*

Proof: The n-colored sequence σ defines a linear ordering $\lambda = v_0, v_1, \dots, v_{n-1}$ of the vertices of G. By using Theorem 5 and Lemma 6 we compute a 3-chain topological book embedding Γ of G such that the linear ordering of the vertices along the spine is λ and each x-monotone chain crosses the spine at most $n-1$ times.

We then replace each spine crossing of Γ with a dummy vertex. Let $\lambda' = w_0, w_1, \dots, w_{n'-1}$ be the left-to-right order of the vertices (dummy or not) along the spine of Γ ($n' \geq n$). Connect w_i to w_{i+1} with a straight-line segment if w_i and w_{i+1} are not adjacent in G ($0 \leq i \leq n'-2$). The resulting planar drawing Γ' describes an augmentation of G with dummy vertices and edges that admits an external Hamiltonian path visiting all real vertices according to σ. Therefore, the path $\mathcal{H}$ from w_0 to $w_{n'-1}$ is an external augmenting n-colored Hamiltonian path of G consistent with σ.

It remains to show that this path induces at most $3n-3$ flat division vertices and at most 2 pointy division vertices per edge. Let e be an edge of Γ and assume that e consists of three x-monotone chains. The number of division vertices along e is the number of spine crossings of e in Γ. By the monotonicity, each spine crossing of an x-monotone chain of e defines a flat division vertex of e. Indeed, a spine crossing w_i of an x-monotone chain π is the common endpoint of two consecutive arcs a_1 and a_2 that share w_i and form an x-monotone portion; this implies that the two endpoints of a_1 and a_2 different form w_i are encountered one before and the other after w_i along the spine of Γ'. Since the order of the vertices along $\mathcal{H}$ is the same as the left-to-right order of the vertices Γ', it follows that w_i is a flat division vertex.

Let w_j be a spine crossing defined by two consecutive x-monotone chains of e. w_j is the common endpoint of two consecutive arcs such that a_1 and a_2 that share w_j and do not form an x-monotone portion. This implies that the two endpoints of a_1 and a_2 different form w_j are encountered both before or both after w_j along the spine of Γ'. Hence w_j is a pointy division vertex.

Since each x-monotone chain of Γ has at most $n-1$ spine crossings and each edge has at most three x-monotone chains, it follows that $\mathcal{H}$ induces at most $3n-3$ flat division vertices and at most 2 pointy division vertices. □

By using Lemma 8 and Theorem 3, we are in the position of proving the following upper bound on the curve complexity of k-colored point-set embeddings.

Theorem 6 *Let G be a k-colored planar graph with n vertices such that $2 \leq k \leq n$ and let S be a k-colored set of points compatible with G. There exists an $O(n^2 \log n)$-time algorithm that computes a k-colored point-set embedding of G on S having curve complexity at most $3n + 2$.*

Proof: Assume first that $k = n$ and let σ be the n-colored sequence induced by S. The algorithm is as follows.

1. Compute the n-colored sequence σ induced by S by sorting the points according to their x-coordinates. Let λ be the linear ordering of the vertices of G defined by σ.
2. By using Lemma 7, compute a simplified 3-chain topological book embedding Γ'' of G such that the left-to-right order of the vertices of G along the spine is λ .
3. By means of Γ'' and by using Lemma 8, compute an external augmenting n-colored Hamiltonian path consistent with σ inducing at most $d_f = 3n-3$ flat division vertices and at most $d_p = 2$ pointy division vertices per edge.
4. By using Theorem 3, construct an n-colored point-set embedding on S such that the maximum number of bends along each edge is $d_f + 2d_p + 1 = 3n + 2$ by using .

If $k < n$, then one can arbitrarily map each vertex of color i $(0 \leq i \leq k-1)$ to a point of S_i (thus defining an n-coloring of G and S) and then use the drawing algorithm just described.

Step 1 can be executed in $O(n \log n)$ time and Step 2 can be executed in $O(n^2 \log n)$ time (see Lemma 7).

Step 3 can be performed as follows. Refer to the data structures in the proof of Lemma 7) and to the technique illustrated in Lemma 8. Visit each arc in the top page of the AVL trees associated with the x-monotone chains of Γ''. Let a be the currently visited arc and let A' be the array that stores all vertices and spine crossings of Γ'', sorted according to their left-to-right order along the spine. By performing a binary search in the array A', we can determine in $O(\log n)$ time whether the endpoints of a are consecutive along the spine. In the affirmative case, we equip the two elements of A' that are the leftmost endpoint and the rightmost endpoint of a with a reference to a. Since there are $O(n^2)$ arcs Γ'', this step can be executed in $O(n^2 \log n)$ time. We now construct the external augmenting n-colored Hamiltonian path by scanning the $O(n^2)$ elements of A' and by connecting with an edge every pair of elements that are not adjacent in Γ''. It follows that the overall time complexity of Step 3 is $O(n^2 \log n)$.

Step 4 is based on the drawing technique of Kaufmann and Wiese [16], that is recalled in Theorem 3. In [16] it is proved that this technique can be executed in linear time for an embedded planar graph with a given external hamiltonian path. Observe that the graph obtained by augmenting G with the edges and the vertices of the external augmenting n-colored Hamiltonian path computed by the previous steps is a planar graph with $O(n^2)$ vertices and has an external hamiltonian path. It follows that Step 4 can be executed in $O(n^2)$ time.

We can therefore conclude that a k-colored point-set embedding of G on S having curve complexity at most $3n + 2$ can be computed in $O(n^2 \log n)$ time. □

7.2 Special Colorings

Since by Theorem 1 k-colored point-set embeddings can have a linear number of edges each requiring a linear number of bends, the upper bound on the curve complexity expressed by Theorem 6 is asymptotically tight. However, there can be special colorings of the input graph that guarantee a curve complexity which depends on k and it does not depend on n.

Let G be a planar graph with n vertices and let λ be a linear ordering of the vertices of G. We present a lemma that studies the relationship between a drawing of an edge in the monotone topological book embedding of G computed by Step (-1) of Algorithm LinearOrderDraw and a simplified 3-chain topological book embedding of G that has the left-to-right ordering of λ. In the statement we say that two vertices u and v of G are consecutive along the spine to mean that there is no vertex of G drawn in the open interval defined by u and v in the spine; clearly, such interval may contain spine crossings.

Lemma 9 *Let Γ be a monotone topological book embedding of G computed by Step (-1) of Algorithm* LinearOrderDraw *and let Γ' be a simplified simplified 3-chain topological book embedding of G computed by the simplification procedure. Let u and v be two vertices of G that are consecutive along the spine of Γ and are consecutive along the the spine of Γ'. Let e be an edge of G such that the drawing of e in Γ does not cross the spine at a point between u and v. Then, also the drawing of e in Γ' does not cross the spine between u and v.*

Proof: Consider the drawing of e in Γ and let v_i, v_j be the endvertices of e with v_i left of v_j. Edge e in Γ' consists of at most three x-monotone chains. More precisely, if the drawing of e in Γ does not cross the spine then, as explained in the proof of Lemma 5, its drawing in Γ' consists of at most two x-monotone chains else it can have also a third x-monotone chain.

Let π be an x-monotone chain of a 3-chain topological book embedding. The *top interval* of π is the union of all intervals of the arcs of π that are in the bottom page. Similarly, the *bottom interval* of π is the union of all intervals of the arcs of π that are in the top page. We shall study the properties of the bottom and top intervals of the x-monotone chains forming e in the drawing computed by Algorithm LinearOrderDraw and then take into account the simplification procedure. Refer also to the notation of Section 6.1.

Assume first that e does not cross the spine in Γ. We will consider next the case that e crosses the spine of Γ. We can partition the vertices of Γ different from v_i and from v_j into three sets: The *backward vertices* are the vertices to the left of v_i along teh spine of Γ; the *in-between vertices* are the vertices in the open interval between v_i and v_j; the *forward vertices* are to the right of v_j. We say that two vertices have the *same type* if they belong to the same partition set and we say that they have *different type* if they do not belong to the same partition set.

From Step 0 to Step $(i-1)$, Algorithm LinearOrderDraw processes the backward vertices and the lower sequence of e does not exist. From Step i to Step $(j-1)$ the in-between vertices are moved to their target positions and the lower sequence of e consists of at most one x-monotone chain that we call π_1. During these steps, when the trajectory of an in-between vertex intersects π_1, the x-monotone chain is modified by creating an arc a in the bottom page. The interval of a contains the target position of the moved in-between vertex plus, possibly, some spine crossings. As a result, at the end of Step $(j-1)$ the top interval of π_1 contains only in-between vertices and spine crossings, while the bottom interval only contains backward vertices, target positions of forward vertices, and spine crossings. At Step j, vertex v_j is moved to its target position; either a second x-monotone chain, that we call π_2, is created or a new arc is added to π_1, depending on the coordinates of the target position of v_j. From Step j to Step $(n-1)$, Algorithm LinearOrderDraw processes the forward vertices by moving them to their target positions. If the trajectory of a forward vertex intersects π_1 or π_2 (or both), the intersected x-monotone chain is modified by creating an arc in the bottom page whose interval contains the target position of the moved forward vertex plus, possibly, some spine crossings. As a result, at the end of Step $(n-1)$ the top interval of π_1 contains only in-between vertices, forward vertices, and spine crossings, while the bottom interval only contains backward vertices and spine crossings. Similarly, the top interval of π_2 contains only forward vertices and spine crossings while its bottom interval contains only backward vertices, in-between vertices, and spine crossings. Hence, the vertices that are in the top interval of π_h $(h = 1, 2)$ and those that are in the bottom interval of π_h have different type.

Consider now the the simplified 3-chain monotone topological book embedding Γ'. As explained in the proof of Lemma 6, a simplified 3-chain topological book embedding is such that the interval of each arc contains at least one vertex of G. Also, a spine crossing of an x-monotone chain of Γ' is a point shared by two arcs belonging to opposite pages. From the discussion above, we have that the intervals of these two arcs contain vertices that have different type. Therefore, an x-monotone chain of Γ' can have a spine crossing between consecutive vertices only if these two vertices have different type.

Now notice that e does not cross the spine of Γ between u and v. This implies that u and v either have the same type or at least one of them is an endvertex of e; in no case however they can have different type. Since u and v are consecutive along the spine of Γ' it follows that e cannot cross the spine of Γ' between u and v.

It remains to study the case that e crosses the spine of Γ. In this case, the vertices of Γ that are not the endvertices of e are partitioned into four sets: The *backward vertices* are the vertices to the left of v_i;

the *in-between vertices of type A* are the vertices in the open interval between v_i and the spine crossing of e; the *in-between vertices of type B* are the vertices in the open interval between the spine crossing and v_j; the *forward vertices* are those to the right of v_j.

The execution of the n Steps of Algorithm LinearOrderDraw can give rise to a drawing Γ_n where e consists of three x-monotone chains. By using a similar analysis as the one of the previous case, one can show that after the simplification procedure is applied, an x-monotone chain can have a spine crossing between consecutive vertices only if these two vertices have different type and conclude that also in this case e cannot cross the spine of Γ' between u and v. □

We are in the position of proving the main result of this section.

Theorem 7 *Let G be a k-colored planar graph with n vertices such that: (i) $1 \leq k < n$; (ii) $|V_i| = 1$ for every $0 \leq i \leq k-2$; (iii) $|V_{k-1}| = n-k+1$. Let S be a k-colored set of points compatible with G. There exists an $O(n^2 \log n)$-time algorithm that computes a k-colored point-set embedding of G on S having curve complexity at most $9k-1$.*

Proof: Since $k < n$, we choose a mapping of the $n-k+1$ vertices of V_{k-1} to the points of S_{k-1}, thus obtaining an n-coloring of G and S. Let σ be the n-colored sequence induced by S and let λ be the linear ordering of the vertices of G defined by σ. Let Γ be the monotone topological book embedding computed at Step (-1) of Algorithm LinearOrderDraw and let Γ' be a simplified 3-chain topological book embedding of G such that the left-to-right order of the vertices along the spine of Γ' is λ (see Lemma 7).

By Lemma 8, G admits an augmenting n-colored Hamiltonian path consistent with σ and inducing at most $d_f = 3n-3$ flat division vertices and at most $d_p = 2$ pointy division vertices per edge. We show that if the mapping between the $n-k+1$ vertices of V_{k-1} and the points of S_{k-1} is chosen in such a way that the order of the vertices of V_{k-1} along ℓ is also maintained along ℓ', then each x-monotone chain of Γ' crosses the spine at most $3k-2$ times. By a reasoning analogous to that of Lemma 8, we can conclude that G admits an augmenting n-colored Hamiltonian path consistent with σ and inducing at most $d_f = 9k-6$ flat division vertices per edge and at most $d_p = 2$ pointy division vertices per edge. Hence, by Theorem 3, G admits an n-colored point-set embedding on S such that the maximum number of bends along each edge is $d_f + 2d_p + 1 = 9k-1$. Clearly, such an n-colored point-set embedding of G on S is also a k-colored point-set embedding of G on S.

Observe that, as described in the proof of Lemma 6, an x-monotone chain of an edge e of Γ' can cross the spine only once between each pair of consecutive vertices. Also, by Lemma 9, there is not a spine crossing if these two consecutive vertices are also consecutive along ℓ and e does not cross ℓ between them. In order to compute an upper bound on the number of spine crossings of an x-monotone chain of Γ', we count the number of consecutive pairs of vertices along the spine of Γ' for which Lemma 9 does not hold. Denote by c_1 the maximum number of pairs of vertices that can be consecutive in ℓ' and not in ℓ; denote by c_2 the maximum number of pairs of vertices u and v such that u and v are consecutive both in ℓ and in ℓ' and e has a spine crossing between u and v in Γ. The wanted upper bound is $c_1 + c_2$.

Since Γ is a monotone topological book embedding, $c_2 = 1$. As for c_1, we observe that the order of the vertices along ℓ' is the same as the order along ℓ except for those vertices of colors $0, 1, \ldots, k-2$. Let v be a vertex having color different form $k-1$ and assume that v is followed and preceded along ℓ by vertices u and w; also assume that v is followed and preceded by vertices u' and w' along ℓ'. Note that the vertices forming pairs $< u', v >$ and $< v, w' >$ are consecutive in Γ' but not in Γ; also the vertices of the pair $< u, w >$ can be consecutive in Γ' but not in Γ. It follows that for every vertex having color different form $k-1$, an x-monotone chain of Γ' can cross the spine at most 3 times, and therefore $c_1 = 3k-3$. Since an edge of Γ' can consist of at most three x-monotone chains, it follows that G admits an augmenting n-colored Hamiltonian path consistent with σ and inducing at most $d_f = 9k-6$ flat division vertices per edge and at most $d_p = 2$ pointy division vertices per edge.

The stated time complexity can be proved by the same analysis in the proof of Theorem 6. □

8 Conclusions and Open Problems

This paper has presented a unified approach to the problem of computing k-colored point-set embeddings of k-colored planar graphs such that the curve complexity of the drawing is optimal. The described results extend and improve known results described in the literature. The used techniques rely on the study of topological properties of planar graphs and on an equivalence relation between computing a k-colored point-set embedding and finding a suitable Hamiltonian path in a graph.

We conclude with some open problems about k-colored point-set embeddings that could be the subject of further research.

1. Reduce the gap between upper and lower bound for the curve complexity of k-colored point-set embeddings.

2. Theorems 1 and 2 show that the total number of bends of k-colored point-set embeddability problem can be quadratic for $2 \leq k \leq n$. It would be interesting to study whether a subquadratic upper can be obtained in the case that the number of points for each color i is cn_i, where c is a constant larger than 1 and n_i is the number of vertices of color i.

3. What is the curve complexity of k-colored point-set embeddings of k-colored trees for small values of k? Notice that the described lower bounds use bi-connected graphs.

Acknowledgements

The authors thank Walter Didimo for useful discussions.

References

[1] M. Abellanas, J. Garcia-Lopez, G. Hernández-Peñver, M. Noy, and P. A. Ramos. Bipartite embeddings of trees in the plane. *Discrete Applied Mathematics*, 93(2-3):141–148, 1999.

[2] J. Akiyama and J. Urrutia. Simple alternating path problem. *Discrete Mathematics*, 84:101–103, 1990.

[3] P. Bose. On embedding an outer-planar graph on a point set. *Computational Geometry: Theory and Applications*, 23:303–312, 2002.

[4] P. Bose, M. McAllister, and J. Snoeyink. Optimal algorithms to embed trees in a point set. *Journal of Graph Algorithms and Applications*, 2(1):1–15, 1997.

[5] U. Brandes and T. Erlebach, editors. *Network Analysis: Methodological Foundations*, volume 3418 of *Lecture Notes in Computer Science*. Springer, 2005.

[6] G. Di Battista, P. Eades, R. Tamassia, and I. G. Tollis. *Graph Drawing*. Prentice Hall, Upper Saddle River, NJ, 1999.

[7] E. Di Giacomo, W. Didimo, G. Liotta, H. Meijer, F. Trotta, and S. K. Wismath. k-colored point-set embeddability of outerplanar graphs. In *Graph Drawing (Proc. GD 2006)*, volume 4372 of *LNCS*, pages 318–329. Springer-Verlag, 2007.

[8] E. Di Giacomo, W. Didimo, G. Liotta, and S. K. Wismath. Curve-constrained drawings of planar graphs. *Computational Geometry*, 30:1–23, 2005.

[9] E. Di Giacomo, G. Liotta, and F. Trotta. On embedding a graph on two sets of points. *International Journal of Foundations of Computer Science, Special Issue on Graph Drawing*, 17(5):1071–1094, 2006.

[10] P. Gritzmann, B. Mohar, J. Pach, and R. Pollack. Embedding a planar triangulation with vertices at specified points. *Amer. Math. Monthly*, 98(2):165–166, 1991.

[11] J. H. Halton. On the thickness of graphs of given degree. *Information Sciences*, 54:219–238, 1991.

[12] A. Kaneko and M. Kano. Straight line embeddings of rooted star forests in the plane. *Discrete Applied Mathematics*, 101:167–175, 2000.

[13] A. Kaneko and M. Kano. Discrete geometry on red and blue points in the plane - a survey. In *Discrete & Computational Geometry*, volume 25 of *Algorithms and Combinatories*, pages 551–570. Springer, 2003.

[14] A. Kaneko, M. Kano, and K. Suzuki. Path coverings of two sets of points in the plane. In J. Pach, editor, *Towards a Theory of Geometric Graphs*, volume 342 of *Contemporary Mathematics*. American Mathematical Society, 2004.

[15] M. Kaufmann and D. Wagner, editors. *Drawing Graphs*, volume 2025 of *Lecture Notes in Computer Science*. Springer-Verlag, 2001.

[16] M. Kaufmann and R. Wiese. Embedding vertices at points: Few bends suffice for planar graphs. *Journal of Graph Algorithms and Applications*, 6(1):115–129, 2002.

[17] J. Pach and R. Wenger. Embedding planar graphs at fixed vertex locations. *Graph and Combinatorics*, 17:717–728, 2001.

[18] K. Sugiyama. *Graph Drawing and Applications for Software and Knowledge Engineers*. Word Scientific, 2002.

[19] R. Tamassia, G. Di Battista, and C. Batini. Automatic graph drawing and readability of diagrams. *IEEE Transactions on Systems, Man, and Cybernetics*, SMC-18(1):61–79, 1988.

Appendix

A1. 2-colored Diamond Graph and Lower Bound

A 2-colored diamond graph is a 2-colored graph as the one depicted in Figure 2(a). More formally, let $n \geq 16$ and let $n' = n - n \mod 16 = 16h$, for a positive integer h; a *2-colored diamond graph* $G_n = (V, E)$ is defined as follows:

- $V = V_0 \cup V_1 \cup V_2$
- $V_0 = \{v_i \mid 0 \leq i \leq \frac{n'}{2} + \left\lceil \frac{n \mod 16}{2} \right\rceil\}$
- $V_1 = \{u_i \mid 0 \leq i \leq \frac{n'}{4} + \left\lfloor \frac{n \mod 16}{2} \right\rfloor\}$
- $V_2 = \{w_i \mid 0 \leq i \leq \frac{n'}{4}\}$
- $E = E_0 \cup E_1 \cup E_2 \cup E_3 \cup E_4$
- $E_0 = \{(v_i, v_{i+1}) \mid 0 \leq i \leq \frac{n'}{2} + \left\lceil \frac{n \mod 16}{2} \right\rceil - 1\}$
- $E_1 = \{(u_i, u_{i+1}) \mid 0 \leq i \leq \frac{n'}{2} + \left\lfloor \frac{n \mod 16}{2} \right\rfloor - 1\}$
- $E_2 = \{(w_i, w_{i+1}),\ (w_{i+1}, w_{i+2}),\ (w_{i+2}, w_{i+3}),\ (w_{i+3}, w_i) \mid 0 \leq i \leq 4h - 1,$ $i \mod 4 = 0\}$
- $E_3 = \{(w_{i+1}, w_{i+4}),\ (w_{i+3}, w_{i+4}),\ (w_{i+1}, w_{i+6}),\ (w_{i+3}, w_{i+6}) \mid 0 \leq i \leq 4h - 5,$ $i \mod 4 = 0\}$
- $E_4 = \{(w_{4h-1}, v_{\frac{n'}{2} + \lceil \frac{n \mod 16}{2} \rceil}),\ (w_{4h-3}, v_0),\ (w_0, u_0),\ (w_2, u_{\frac{n'}{4} + \lfloor \frac{n \mod 16}{2} \rfloor})\}$

Let S be an alternating bi-colored sequence compatible with G_n and let $p_0, \ldots, p_{n-1}$ be the points of S ordered according to their x-coordinates. Let Γ_n be a 2-colored point-set embedding of G_n on S where z_i is the vertex of G_n that is mapped to p_i. In contrary to the k-colored case ($3 \leq k \leq n$), z_i and z_{i+1} can be adjacent in Γ_n. This can happen at most twice since only the vertices w_{4h-3} and w_{4h-1} are adjacent to a vertice of set V_0. Connect in Γ_n z_i and z_{i+1} with a straight-line segment ($i = 0, \ldots, n-2$); the obtained path is called *bi-colored path* Π *on* Γ_n.

Lemma 10 *Let G_n be a 2-colored diamond graph and let S be an alternating bi-colored sequence compatible with G_n. Let Γ_n be a 2-colored point-set embedding of G_n on S and let Π be the bi-colored path on Γ_n. Π crosses at least $\frac{n'}{8} - 1$ edges of Γ_n, where $n' = n - (n \mod 16)$; also, Π crosses each of these edges at least $\frac{n'}{8}$ times.*

Proof: We will use the definition of a cycle $C \in G_n$ that separates a subset of vertices from another subset of vertices which was already explained in the proof of Lemma 2. In every planar drawing of G_n each of the h cycles defined by the edges in set E_2 separates all vertices in V_0 from all vertices in V_1. In the same way, each of the $h-1$ cycles defined by the edges in set E_3 separates all vertices in V_0 from all vertices in V_1. Let $n'' = n - n' = n \mod 16$. As we have $\frac{n'}{4} + \lfloor \frac{n''}{2} \rfloor$ vertices in the interior region defined by the cycles C and $\frac{n'}{4} + \lceil \frac{n''}{2} \rceil$ in the exterior region defined by these cycles, each cycle is crossed $\frac{n'}{2} + n'' - 1$ times. Since each cycle has four edges, we have that at least $2h - 1 = \frac{n'}{8} - 1$ edges are crossed at least $\lceil \frac{n'}{8} + \frac{n''}{4} - \frac{1}{4} \rceil \geq \lceil \frac{16h}{8} - \frac{1}{4} \rceil = \lceil 2h - \frac{1}{4} \rceil = 2h = \frac{n'}{8}$ times. $\square$

By means of Lemma 1 and Lemma 10 the following lower bound for 2-colored point-set embeddings can be proved.

Theorem 2 *For every $n \geq 16$ there exists a 2-colored planar graph G_n with n vertices and a 2-colored set of points S compatible with G_n such that any 2-colored point-set embedding of G_n on S has at least $\frac{n'}{8} - 1$ edges each having at least $\frac{n'}{8} - 1$ bends, where $n' = n - (n \mod 16)$.*

Proof: Given any $n \geq 16$ construct a 2-colored diamond graph G_n. Let S be an alternating bi-colored sequence compatible with G_n. Let Γ_n be a 2-colored point-set embedding of G_n on S and let Π be the bi-colored path on Γ_n.

By Lemma 10 there are at least $\frac{n'}{8} - 1$ edges of Γ_n that are crossed by Π at least $\frac{n'}{8}$ times. By Lemma 1 each of these edges has at least $\frac{n'}{8} - 1$ bends. □