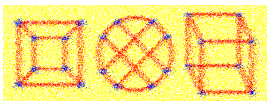

Journal of Graph Algorithms and Applications
http://jgaa.info/ vol. 30, no. 2, pp. 199–235 (2026)
DOI: 10.7155/jgaa.v30i2.3273

Visualizing Treewidth

Alvin Chiu[1] *Thomas Depian*[2] *David Eppstein*[1] *Michael T. Goodrich*[1] *Martin Nöllenburg*[2]

[1]University of California, Irvine, USA
[2]TU Wien, Vienna, Austria

Submitted: October 2025 Accepted: August 2026 Published: August 2026

Article type: Regular paper Communicated by: V. Dujmović, F. Montecchiani

Abstract. A *witness drawing* of a graph is a visualization that clearly shows a given property of a graph. We study and implement various drawing paradigms for witness drawings to clearly show that graphs have bounded pathwidth or treewidth. Our approach draws the tree decomposition or path decomposition as a tree of bags, with induced subgraphs shown in each bag, and with "tracks" for each vertex of the graph connecting its copies in multiple bags. Within bags, we optimize the vertex layout to avoid crossings of edges and tracks. We implement a visualization prototype for crossing minimization using dynamic programming for graphs of small width and heuristic approaches for graphs of larger width. We explore the design space for width-witness drawings and investigate drawing styles that render the subgraph for each bag as an arc diagram with one or two pages or as a circular layout with straight-line edges, and we render tracks either with straight lines or with orbital-radial paths. Finally, we report results from an expert evaluation assessing different witness drawing styles.

1 Introduction

Work in graph drawing can be often viewed as studying paradigms for visualizing a graph, G, to clearly illustrate one or more properties of G while also optimizing one or more quality criteria, such as area or number of edge crossings; see, e.g., [19, 23, 28, 45, 68]. This viewpoint is related to the concept of a *visual proof*, which is a proof given as a visual representation called a *witness* (or visual certificate); see, e.g., [40, 57–59]. Ideally, a visual proof should be clear and concise: the property being proven should be immediately discernible simply by examining the witness.

Special issue on Selected papers from the Thirty-third International Symposium on Graph Drawing and Network Visualization, GD 2025

T. Depian and M. Nöllenburg acknowledge support from the Vienna Science and Technology Fund (WWTF) [10.47379/ICT22029]. M. T. Goodrich and D. Eppstein acknowledge support from NSF grant 2212129.

E-mail addresses: chiua13@uci.edu (Alvin Chiu) tdepian@ac.tuwien.ac.at (Thomas Depian) eppstein@uci.edu (David Eppstein) goodrich@uci.edu (Michael T. Goodrich) noellenburg@ac.tuwien.ac.at (Martin Nöllenburg)

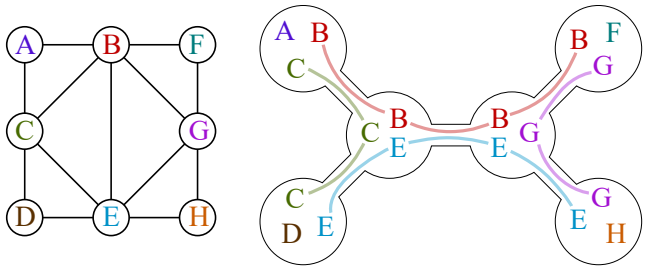


Figure 1: A witness drawing of a graph with treewidth 3. Tracks are shown color coded to illustrate the support for each vertex in the graph. Note that this visualization does not draw graph edges inside the bags of the tree decomposition. Public domain images by David Eppstein [31].

A prominent example is a crossing-free drawing as a witness for planarity. In this paper, we are interested in visualizing treewidth.

A *tree decomposition* of a graph, $G = (V, E)$, is a tree, T, with nodes, $V_1, V_2, \ldots, V_k$, which are called *bags*, where

1. Each V_i is a subset of V and $\bigcup_{i=1}^{k} V_i = V$.
2. If $v \in V_i$ and $v \in V_j$, then v is in each bag in the (unique) path in T from V_i to V_j.
3. For each edge $(v, w) \in E$ there is a bag that contains both v and w.

The *width* of a tree decomposition is one less than the size of its largest bag, and the *treewidth* of a graph is the smallest width of a tree decomposition. The *pathwidth* of a graph is the width of a smallest-width tree decomposition whose tree is a path. See, e.g., [14, 25, 26, 29, 48, 54] for more details regarding these topics, including applications in graph drawing and results that many NP-hard optimization problems can be solved in polynomial time for graphs with bounded treewidth or pathwidth.

We are interested in paradigms for witness drawings of graphs to certify their bounded pathwidth or treewidth. Our approaches are inspired by an example visual proof from Wikipedia that a graph has treewidth 3; see Figure 1. In this illustration, each bag is drawn as a disk with its member vertices drawn as points inside the disk. In addition, for each vertex, $v \in V$, in this drawing, there is a set of curves, each of which we call a *track*, that connect the different copies of v in the bags of the tree decomposition, i.e., the *support* of v in T. The set of tracks in T for a vertex $v \in V$ in such a drawing will form a subtree in T, which will always be a path if T is itself a path. Note, however, that even if the support for every vertex is a path, this does not imply that T is a path, e.g., as shown in Figure 1.

To provide a witness drawing for the pathwidth or treewidth of a graph, $G = (V, E)$, we add one more requirement to our visualization, which is missing from the visualization shown in Figure 1, namely *information faithfulness*, i.e., that the drawing represents the (entire) graph G and its decomposition T. In particular, we require that for each edge, $(v, w) \in E$, we must draw (v, w) as a curve inside each bag V_i such that $v, w \in V_i$ in the given tree decomposition, T, of G. Still, we are interested in such witness drawings that minimize edge and track crossings, as in Figure 1. Note that we want the witness drawing to be information faithful on its own. In particular, when requiring information faithfulness, we do not consider combined drawings of G and T as in Figure 1.

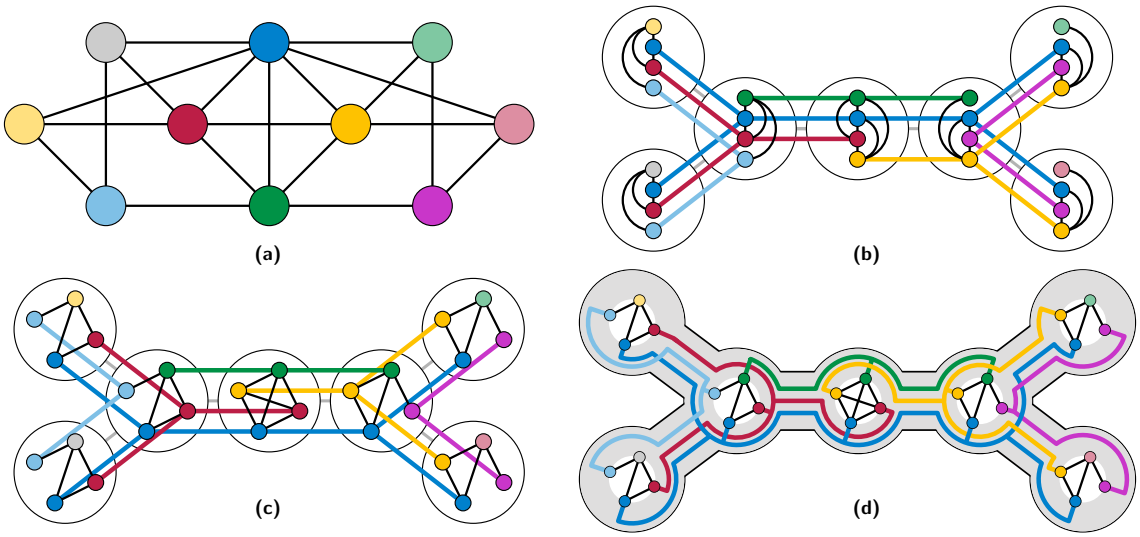


Figure 2: Witness drawings for a graph, G, with treewidth 3. A standard drawing of G is shown in **(a)**. We show in **(b)** a witness drawing of a tree decomposition, in which the subgraph for each bag is drawn as an arc diagram. The two lower witness drawings show the subgraph for each bag in a circular layout with straight-line edges. In **(c)**, the tracks connecting instances of the same vertex are drawn as straight segments, while in **(d)** the track edges are routed in orbits surrounding the bag vertices. The track edges are colored to match the vertices they represent. The drawing in **(b)** has 6 track-track crossings, no edge-edge crossings, and 6 track-edge crossings. The drawing in **(c)** has 4 track-track crossings, 1 edge-edge crossing, and 13 track-edge crossings. The drawing in **(d)** has 12 track-track crossings, 1 edge-edge crossing, and no track-edge crossings.

To see why this requirement is important, consider a graph G with pathwidth at most ω. Its *interval-width* is bounded by $\omega+1$, i.e., G is a subgraph of an interval graph G' whose largest clique has size at most $\omega + 1$ [18]. Since interval graphs can be represented as closed intervals on the real line, such a representation of G' can be seen as a certificate that G has bounded pathwidth. However, we argue that this is not an information faithful witness drawing as it shows G' rather than G and, in particular, does not allow a reconstruction of G from the (interval) drawing of G'.

Our Results. In this paper, we systematize and explore witness drawing styles for graphs with bounded treewidth or pathwidth. We focus on styles that, given a graph, $G = (V, E)$, draw the vertices in each bag of a tree decomposition, T, of G as points in a disk, either in a circular layout or a linear layout. More precisely, for each vertex, $v \in V$, we connect each pair of copies of v in adjacent bags in T with a curve we call a *track*, ideally with all the tracks color coded with the same color (the color for v). For each bag, V_i, in T, we draw the edges of the subgraph of G induced by V_i. In the case of a linear layout of vertices, we draw this subgraph as an arc diagram [73] (either as one-page or two-page book drawing) and in the case of a circular layout of vertices, we draw this subgraph as a circular layout with straight-line edges [10]. Our optimization goal is to minimize the total number of edge crossings, including track-track crossings, edge-edge crossings, and track-edge crossings. See Figure 2 for three different witness drawing styles. We design fixed-parameter-tractable linear-time crossing minimization algorithms for graphs with bounded

pathwidth or treewidth. We implement the above-mentioned algorithm for drawings with a linear layout, suggest a faster heuristic approach, and compare these algorithms. Moreover, we conduct an expert evaluation to collect preliminary feedback on the suggested drawing styles and identify the most promising directions for future work.

Related Work. Mehlhorn *et al.* [59] and McConnell *et al.* [57] introduce the paradigm of *certifying algorithms*, which output their result as well as a concise proof, called a "witness", that shows that the algorithm produced this output correctly. Subsequent to this work, there has been considerable work on certifying algorithms, including their inclusion in the LEDA and CGAL systems [44]. For example, additional work for witness drawings in graph drawing include work on visualizing proximity properties, such as for Delaunay graphs, Gabriel graphs, and rectangle graphs; see, e.g., [3–6, 50–53].

In GD 2024, the *GraphTrials* framework was introduced, which involves a visual proof for a graph property that can be used in an interactive proof modeled after a bench trial before a judge where a prover establishes that a graph has a given property based on its visualization [36]. Although the authors did not include pathwidth or treewidth in their GraphTrials framework, we feel that our approach to producing witness drawings for graphs with bounded pathwidth or treewidth nevertheless fits into their GraphTrials framework. In particular, we believe that our witness drawings satisfy the three required properties for visual certificates as defined in the framework [36, 40]. More concretely, our witness drawings are (i) *unimpeachable*, i.e., information faithful and allow a judge to perceive enough information to validate the property, which is the case since the visualization inside the bags together with the tracks provides all the information to ensure that the decomposition at hand is valid. Moreover, they are (ii) *checkable*, i.e., it is easy and effortless for a judge to validate the property, as they facilitate the formation of a mental model. In particular, exploiting the Gestalt principles of *Common Region*, *Proximity*, and *Similarity* [71], the mental model might, on the one hand, clearly partition the drawing into smaller drawings for different subgraphs, while, on the other hand, at the same time linking occurrences of the same vertex in different bags. Although the task of checking the certificate is not an instantaneous process even when using this mental model, we argue that it is still faster and easier than not having a witness drawing (and the resulting mental model). Finally, a visual certificate must be (iii) *simple*, i.e., allow a judge to derive all conclusions, in our case bounded pathwidth or treewidth, entirely from the mental model. Since the above-suspected mental model contains all components of a decomposition, this is indeed the case.

We are interested in the natural optimization criterion of minimizing edge crossings in a witness drawing for a graph with bounded pathwidth or treewidth, including track-track crossings, edge-edge crossings, and track-edge crossings. In particular, few track-track crossings can aid in verifying that our decomposition satisfies Property 2, i.e., that the bags containing the vertex v induce a connected subgraph of T, for all $v \in V$. Furthermore, minimizing edge-edge and track-edge crossings improves the visual quality of the witness drawing and can thus support the verification of the (remaining) properties. Although we are not aware of any prior work on this criterion, minimizing track-track crossings is related to minimizing crossings in storyline drawings [24, 42, 49, 69, 70] and metro maps [2, 7, 11, 12, 35, 61–63]. Minimization of edge-edge crossings within the bags is related to crossing minimization in linear and circular graph layouts [8, 10, 27, 41, 47, 55, 73].

Multiple past works include individual visualizations of tree decompositions. These include drawings of a tree with each bag shown as a set of vertices within each tree node, separate from any drawing of the graph [14, 54], and drawings of a graph overlaid by bags drawn as regions that surround or connect the vertices [32]. Separately from their applications in treewidth and

structural graph theory, tree decompositions with bags of non-constant size have also been used in other ways: for instance, SPQR decompositions are, essentially, tree decompositions of adhesion 2, whose bags induce 3-connected subgraphs, and these have often been visualized as a tree of 3-connected components [20, 60]. However, we are unaware of past works studying visualizations of tree decompositions in any systematic way.

2 Preliminaries

For an integer $p \geq 1$, we use $[p]$ as a shorthand for the set $\{1, 2, \ldots, p\}$. Let $G = (V, E)$ be a graph with vertex set V and edge set E. All considered graphs are simple and undirected. Throughout this paper, we use $n := |V|$ and $m := |E|$ to refer to the number of vertices and edges of G, respectively. For a tree (or path) decomposition T of G with bags $V_1, V_2, \ldots, V_k$, we let ω_T, or simply ω if T is clear from the context, denote its width. Recall $\omega_T := \max_{i \in [k]} |V_i| - 1$. We use $\overline{\omega} := \omega + 1$ as a shorthand for the maximum number of vertices in a bag. Without loss of generality, we root the tree T at an arbitrary bag. We use *decomposition* as a synonym for path and tree decompositions. Furthermore, we assume that the decomposition T consists of $k = \mathcal{O}(n)$ bags and that each bag of T has degree at most three. It is well-known that every graph of treewidth ω admits a certifying tree decomposition with $k = \mathcal{O}(n)$ fulfilling this property [15, 56]. In particular, in *nice tree decompositions*[1], which have found attention in the design of parameterized (dynamic programming) algorithms [18], every bag has degree at most three in T. With slight abuse of notation, we write $V_i \in T$ to indicate that V_i is a bag of T. Furthermore, for a bag $V_i \in T$, we let $G_i := G[V_i]$ and $E_i := E(G_i)$ denote the subgraph of G induced by V_i and its edge set, respectively.

3 Exploring the Design Space of Width-Witness Drawings

Before we describe how to compute a witness drawing for a decomposition T of G, we first take a closer look at the different witness drawings from Figures 1 and 2. To that end, we first explore the design space of width-witness drawings and categorize different drawing styles based on their features; recall that we focus on witness drawings for T and not on combined drawings of T and G. Afterwards, we elaborate on three particular drawing styles that we discuss in the upcoming sections in more detail. We emphasize that the following exposition, also in view of the plethora of different graph visualization styles, is merely an exploration of the design space. The presented drawing styles should be seen as a suggestion and inspiration; styles that we do not mention in this work explicitly can still be sensible.

A witness drawing Γ of a given decomposition T of G consists of a drawing of (i) the tree (or path) T, (ii) the subgraph G_i in each bag $V_i \in T$, and (iii) the support for each vertex of G, i.e., the tracks. Consequently, also the design space features one dimension for each of the above-mentioned design choices, i.e., how to draw the tree (or path) T, the subgraphs G_i in the bags, and the tracks. Moreover, when computing a witness drawing Γ, we seek to optimize for a certain (iv) quality criterion. Note that the drawing Γ might contain crossings, and we differentiate between the following three crossing types; recall also Figure 2: If two tracks cross, we call it a track-track crossing (or simply t/t-crossing), if a track and an edge in a disk cross, it is a track-edge (t/e) crossing, and if two edges of G cross inside a disk, we call it an edge-edge (e/e) crossing. If neither

[1] In nice tree decompositions, T is rooted and every bag V_i is either an *introduce bag*, where $V_i = V_j \cup \{v\}$ with V_j being the single child of V_i and $v \notin V_j$, a *forget bag*, where $V_i = V_j \setminus \{v\}$ with V_j being the single child of V_i and $v \in V_j$, or a *join bag*, where $V_i = V_j = V_k$ and V_j and V_k are the only two children of V_i [18].

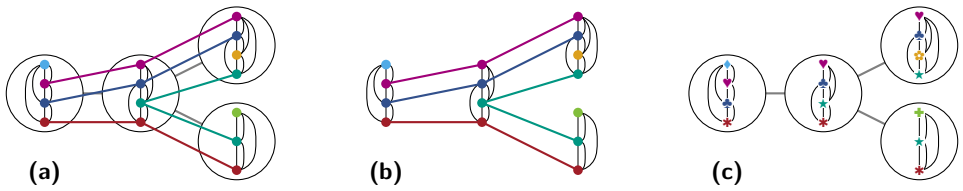


Figure 3: A witness drawing of a tree decomposition where we **(a)** draw the tree T and the tracks, **(b)** draw only the tracks, and **(c)** draw only the tree and use different glyphs instead of the tracks.

of these crossings exist, we call Γ *planar*. Note that we do not count crossings with edges of T. The four parameters mentioned above define the design space. In the following, we explore for each of the four dimensions different choices and discuss their potential influence on the obtained witness drawing. We refer to Section 3 for an overview of possible drawing styles.

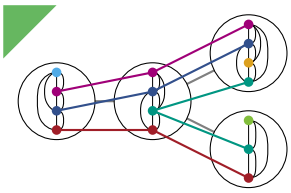
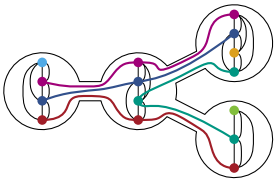
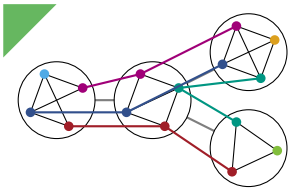
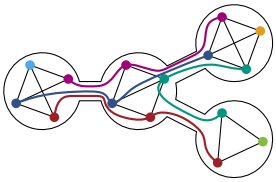
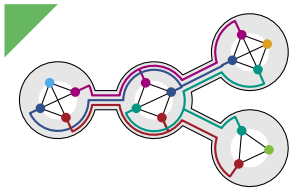
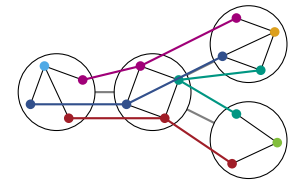
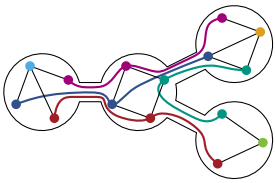
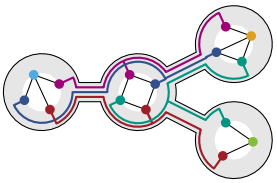
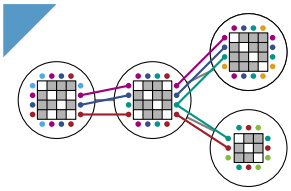
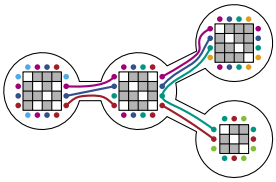
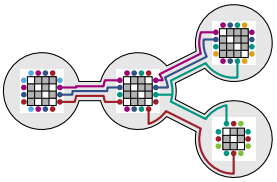
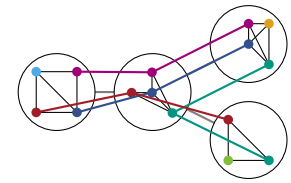
Drawing Style of the Tree T. The first dimension of the design space centers around how the underlying tree (or path) T itself is visualized, i.e., the drawing of its bags and edges. The bags $V_i \in T$ can be drawn using geometric shapes such as disks or squares. The selected shape should fit the chosen style for the drawing of the subgraphs G_i. To avoid emphasizing some bags over others, we can draw all bags using the same shape and in the same size. Alternatively, if we want to emphasize bags containing many vertices, we can scale the bag proportionally to the number of vertices it contains. As an entirely different visualization style, we can completely omit the contour of the individual bags; compare also Figures 3a and b. In this case, we rely on the Gestalt principle of proximity [71] to group together the drawings of subgraphs into the bags. For the edges of T, we can decide to draw them explicitly, e.g. using straight lines, or omit them and indicate adjacency via the relative positioning of the bags in the plane. Finally, there are many different ways to layout the tree T itself [67]. Since we want to enrich the bags with further information, a node-link representation of T should be chosen. Moreover, we should draw T without crossings, e.g., rightward planar or orthogonal planar. In case of a rooted tree, e.g., for nice tree decompositions, which are rooted [18], the embedding should clearly reflect the root, e.g., by using a downward- or rightward-planar embedding.

Drawing Style of the Subgraphs G_i. The second dimension of the design space influences the visualization of the subgraph G_i in each bag $V_i \in T$. While the suitability of a visualization style depends on the structure of both the graph G_i and the tree T, it is desirable to represent the graphs consistently across bags. Moreover, if possible, we should aim for an even distribution of the vertices within each disk.

Consider again Figure 1. There, the vertices V_i are arranged along a circle inside the bag. If we now draw the edges as straight-line chords of said circle, the resulting drawing is, combinatorial speaking, equivalent to a one-page book drawing of G_i [13]. Instead of straight chords, we can also draw the edges using arcs, similar to chord diagrams and the chordlink model [1]. Alternatively, we can visualize the adjacencies using drawing styles that are inspired by (outer-)confluent drawings [21, 22, 33, 34, 38, 39]. The former makes more use of the space inside bags and can, for example, highlight highly connected vertices.

Instead of a circular arrangement, we can also arrange the vertices along a vertical or horizontal line within the bag. This results in a one- or two-page book drawing of G_i, where we draw the

Table 1: Visualization of a subset of the design space. In all drawing styles, we draw each bag $V_i \in T$ as a disk D_i of uniform radius and T rightward planar. In a consistent embedding, the embedding inside each bag is identical to that of a drawing of the graph, implying that identical vertices are placed at the same relative position in different disks. The gray region indicates the dedicated track routing area. Combinations marked with a cross are not applicable since the existence of a dedicated interface curve, along which all the vertices in a bag are placed and which separates the track routing area from the bag area cannot be guaranteed. We indicate in green design styles that we study algorithmically in this paper and in blue those that we additionally consider in our expert evaluation in Section 8.

Graphs	Tracks: Straight	Tracks: Smooth Curves	Tracks: Track Routing Area
Book Drawing			
Circular Placement			
Edges Drawn Only Once			
Incidence Matrix			
Consistent Embedding			

edges as arcs to either side of the vertex line [47, 73]. For a one-page book drawing, all arcs are either to the right or to the left of the vertices, but this can change between different bags. In two-page book drawings, the arcs can be on both sides of the vertices.

For dense graphs, adjacency matrices might be the preferred choice; compare also with the NodeTrix model [43]. We believe that this representation, as well as book drawings, are particularly suited for path or tree decompositions in which each bag has low degree in T. The reason for this is that in these styles, all vertices are arranged along a line inside the bags which gives four (or two) natural attachment directions for the tracks that interfere little with the remaining drawing.

Moreover, we can also use force-based approaches [37] to draw G_i or require that identical subgraphs are drawn consistently across bags. In essence, any reasonable graph visualization or representation, as long as it exists for each subgraph, falls into this dimension of the design space. Specific drawing styles that do not exist for all graphs, e.g., plane drawings, cannot be used since we must provide a drawing for the subgraph of each bag.

Finally, recall that a tree decomposition requires only that for every edge $uv \in G$ there is *some* bag that contains the vertices u and v. Hence, instead of representing uv in all such bags, we can also decide to do so in just one of them. This might reduce clutter in the individual drawings, but at the cost of information faithfulness: the drawing might then only represent a subgraph of G_i.

Drawing Style of the Tracks. The third dimension of the design space concerns the drawing of the tracks that link occurrences of the same vertex in different bags. The most natural way is to connect them on the shortest path possible using straight lines. However, we can also try to match the structure of the tree decomposition by using smooth curves as in Figure 1 (in which case the edges of the tree T should accommodate the tracks). If tracks are drawn as straight lines, they will be monotone with respect to the tree layout and we enforce the same for smooth curves. A further option would be to not draw the tracks at all to reduce the clutter of the drawing. Instead, we could rely on visual encodings, such as color or shape, to convey that vertices in different bags belong together; compare also Figures 3a and c. Finally, we might want to restrict the way the tracks can connect to the vertices in a bag. One way to do this is to introduce a dedicated (and visually separated) track routing area inside the bags, which additionally prevents t/e-crossings and helps in reducing the clutter around the drawing of G_i, for each $V_i \in T$. However, note that we can only use a track routing area if we can define for each bag $V_i \in T$ a continuous interface curve between the track routing area and the drawing of G_i that intersects all vertices but no edges or tracks, i.e., precisely the vertices V_i have access to the interface curve.

Optimization Criteria. The final dimension concerns the quality criteria used to evaluate and compare different witness drawings. Here, classical readability criteria from graph drawing can be applied, e.g., minimizing the number of crossings or the length of the tracks [46, 65]. When minimizing crossings, we can either treat all types of crossings equally or prioritize certain types over others. For example, when tracks are colored, crossings between them could be considered less problematic than crossings between edges of the graph. Another possible optimization criterion could be to minimize the bag area covered by tracks. Thereby, we improve the visibility of the drawings of G_i for each $V_i \in T$. Moreover, depending on the drawing style of the subgraphs G_i or the tracks, further quality criteria such as the number of bends or the crossing angle could be relevant. To identify the priority of these optimization criteria or suggest alternative ones user studies can be conducted.

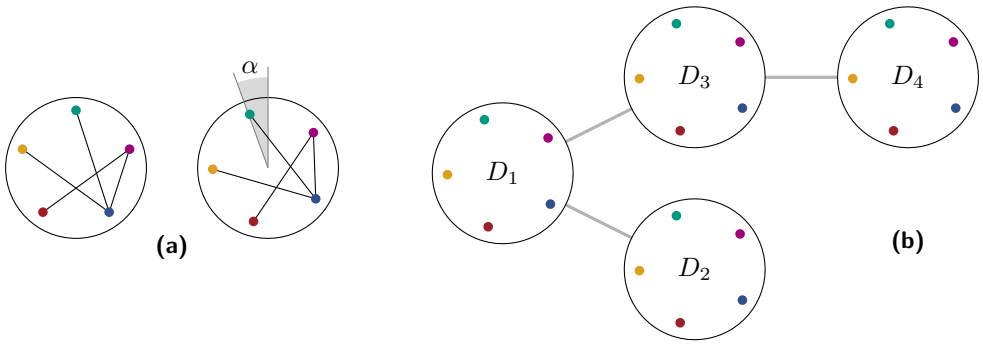


Figure 4: **(a)** The same circular arrangement of V_i can result in different vertex placements in the disk D_i depending on the angle α: left: $\alpha = 0$, right: $\alpha = 20°$. **(b)** Left-to-right drawing of the tree T with four bags. We omit visualizing edges and tracks to maintain readability.

Further Aspects of Width-Witness Drawings. Observe that most of the above-discussed design choices have an algorithmic impact on the computation of witness drawings. However, a comprehensive characterization of the design space of width-witness drawings should also include rendering-related dimensions such as line width or the used color palette; see also the book by Ware [72] for further information on these aspects.

Studied Drawing Styles. Section 3 demonstrates different combinations of drawing styles for the tracks and the subgraphs. In each of them, we explicitly visualize the underlying tree T by drawing it rightward-planar, visualizing each bag as a disk, and minimize the number of crossings. A detailed algorithmic discussion of all drawing styles from Section 3 is beyond the scope of the paper. Instead, we focus on a subset of the styles that we deem particularly relevant for an initial study of crossing minimization in width-witness drawings. On the one hand, these are styles that depict the graph within each bag as a book drawing due to the algorithmic advantages of their combinatorial nature. On the other hand, these are styles with a circular vertex placement, since they are also frequently used in existing visualizations; see, for example, Figure 1. In contrast, we regard the representation of tracks with smooth curves mainly as a postprocessing step, and, therefore, do not consider it in our algorithms.

The drawing styles for which we present algorithms in this paper are highlighted in green in Section 3. We illustrate them with a larger example in Figure 2 and describe them in more detail in the following. We use parenthesized letters to refer to each style. In all studied drawing styles, each bag $V_i \in T$ is represented as a disk D_i of uniform radius. Moreover, we draw T explicitly rightward planar, i.e., the leftmost disk corresponds to the root of T and children of each bag are evenly distributed to their right. The edges of T are visualized as a straight line; see also Figure 4b.

Linear Drawings (L). A *linear drawing*, as in Figure 2b, consists of a one- or two-page book drawing per bag [47, 73]. The tracks for each support are straight-line edges connecting the respective vertices. We use L1 and L2 to differentiate between one- and two-page book drawings where needed.

Circular Drawings (C). Figure 2c shows a *circular drawing* of T. There, we arrange the vertices V_i along a circle inside the disk D_i and draw the edges E_i as straight-line chords of said

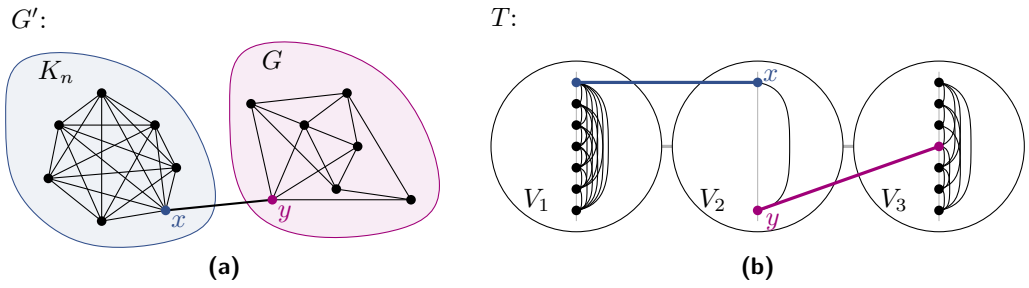


Figure 5: **(a)** The graph G' with $2n$ vertices and a **(b)** path decomposition T of G' of width $n-1$.

circle. Recall that, from a combinatorial perspective, this is equivalent to a one-page book drawing of G_i [13]. As for linear drawings, tracks are straight lines.

Orbital Drawings (O). The last drawing style that we investigate, *orbital drawings*, is inspired by the edge routing in previous drawings with circular vertex placements [9, 16]. It is illustrated in Figure 2d and uses again a circular arrangement of the vertices. However, we now explicitly introduce a *track-routing area* in each disk, which consists of the remaining part outside, i.e., around, the drawing of G_i. The tracks are aligned as orbits inside the track-routing area that can be routed clockwise or counterclockwise around D_i. Observe that this drawing style has no track-edge crossings and all track-track crossings occur inside the track-routing area. While this can result in a cleaner visualization, forcing tracks to stay inside the track-routing area may clutter drawings of decompositions of larger width.

Note that for linear drawings, the vertex permutation uniquely defines each vertex' position inside the disk, but for the other two drawing styles this still leaves the rotation of the drawing as a degree of freedeom. To that end, we let $0 \leq \alpha < 2\pi/\overline{\omega}$ denote the starting angle of the first vertex inside each disk, where $\alpha = 0$ corresponds to a placement at twelve o'clock (Figure 4a). We choose a single α across all disks, and we select it so that no interior of a track passes through a vertex independently of the order of the vertices inside each disk.

4 Minimizing Crossings in Witness Drawings is NP-hard

Before we present our fixed-parameter algorithms for computing crossing minimal witness drawings of graphs with bounded path- or treewidth, we justify the focus on decompositions of small width.

Theorem 1. *Deciding if a path decomposition T of a graph G has an L1-drawing with at most c crossings is NP-hard. The same holds true for L2-, C-, and O-drawings.*

Proof: First, we show the statement for L1-drawings and later argue that it also holds for the other drawing styles. We reduce from the NP-complete problem One-Page Crossing Minimization, which asks for a given graph G whether it admits a one-page book drawing with at most c crossings [55].

Let (G, c) be an instance of One-Page Crossing Minimization with $G = (V, E)$ and $n = |V|$. We construct the graph $G' = (V', E')$ and the corresponding path decomposition T as follows. For G', we take the disjoint union of the two graphs G and $K_n = (V^n, E^n)$, i.e., a complete graph on n vertices. Observe $|V'| = 2n$. Let x be an arbitrary vertex of K_n and y an arbitrary vertex

of G; we add to G' the edge (x, y). As G' contains a clique on n vertices, its treewidth (and thus pathwidth) is at least $n-1$ [18]. We now construct the path decomposition T. In particular, we introduce the bags $V_1 = V^n$, $V_2 = \{x, y\}$, and $V_3 = V$, connected in this order. Observe that this shows that the treewidth (and pathwidth) of G' is at most $n-1$. Recall that G' contains, by construction, K_n as subgraph, which has a pathwidth (and treewidth) of $n-1$. Therefore, T is an optimal decomposition of G' with respect to its width. It is known that a one-page book drawing of K_n has $\binom{n}{4}$ crossings [66], and, due to symmetry, any linear order of K_n's vertices induces this many crossings. We now use this fact to show that (G, c) has a one-page book drawing with at most c crossings if and only if T has an L1-drawing with at most $\binom{n}{4} + c$ crossings.

For the forward direction, let $\prec_G$ be a linear order of V that yields a one-page drawing of G with at most c crossings. We construct a drawing Γ of T by using an arbitrary order $\prec_1$ on V_1 that starts with x, i.e., $x \prec_1 v$ for every $v \in V^n$, $v \neq x$, and set $x \prec_2 y$ and $\prec_3 = \prec_G$; see also Figure 5. Observe that this drawing has no t/t- and t/e-crossings. Thus, there are only e/e-crossings: $\binom{n}{4}$ in the first, and c in the third bag. For the backward direction, assume that T admits an L1-drawing Γ with at most $\binom{n}{4} + c$ crossings. Without loss of generality, we can assume that the bags V_1, V_2, and V_3 are drawn in this order from left to right. The drawing of K_n in the first bag has $\binom{n}{4}$-many e/e-crossings. Thus, any drawing of G in the third bag can have at most c e/e-crossings. We can use its linear order $\prec_3$ as a witness for (G, c).

The above arguments establish NP-hardness for L1-drawings. One-page book drawings are equivalent to straight-line drawings where the vertices are placed in convex position [13, 30]. Furthermore, between each pair of adjacent bags there is only one track edge. Therefore, the same construction and reasoning yields also NP-hardness for C- and O-drawings independent of the value for α. For L2-drawings, we can use the known NP-hardness for determining whether a graph has a two-page book embedding, i.e., a crossing-free two-page book drawing. Using the facts that the two-page crossing number of K_n is known [66] and that book drawings are agnostic to cyclic rotations of the linear order $\prec_G$, we also obtain hardness for L2-drawings. □

Since a path decomposition is a restricted form of a tree decomposition, Theorem 1 also extends to the latter type of decompositions.

5 FPT-Algorithms for Bounded-Width Decompositions

We now focus on the more restrictive (but arguably also more practical) setting where we want to compute a witness drawing of a path decomposition T of G of bounded width ω. In particular, we show that computing crossing-minimal witness drawings is fixed-parameter tractable in ω. In Sections 5.1 and 5.2, we first present dynamic programming (DP) algorithms for L-drawings of path- and tree-decompositions, respectively. In Section 5.3, we discuss how to adapt these DP algorithms for C- and O-drawings, i.e., with circular vertex placement.

Throughout this section, we only consider *combinatorial crossings*, i.e., crossings that are purely defined by the relative position of vertices in bags and the bags themselves. In particular, we neglect crossings that can be avoided by an appropriate placement of the bags and their included vertices.

5.1 Linear Drawings for Path Decompositions

We first consider linear drawings, which visualize each G_i as a book drawing with one or two pages and draw the tracks as straight lines. Observe that the drawing of G_i is uniquely defined by the linear order $\prec_i$ of V_i and the function $\sigma_i \colon E_i \to \{\ell, r\}$, also called *page assignment*, that assigns

each edge to the left or right page, respectively. Throughout this section, we let the bags of T be ordered as they appear on the path T and assume V_1 to be the leftmost bag.

All of our algorithms, including those for C- and O-drawings, are based on the insight that, for each bag $V_i \in T$, the number of crossings involving edges and/or tracks incident to V_i is determined by the drawings of G_i in the disk D_i and the drawings of the induced subgraphs of neighboring bags. This enables us to employ a dynamic programming (DP) algorithm which processes the path decomposition from left to right and computes a crossing-minimal drawing for the first i bags, for every $i \in [k]$. Since the framework of our algorithms is identical across all drawing styles (and decomposition types), we first describe its general structure and later discuss how it can be adapted to specific drawing styles and tree decompositions.

A General DP Algorithm. We let k denote the number of bags in T and recall $\overline{\omega} = \omega + 1$. Let Γ_i be a drawing of the graph G_i in the disk D_i for each $i \in [k]$. Let $\mathcal{G}_i$ denote the set of all possible drawings Γ_i of G_i. All drawing styles admit a combinatorial representation of Γ_i, which implies that $\mathcal{G}_i$ is finite for each $i \in [k]$. Let $\Delta(\Gamma_i)$ denote a drawing of the tracks inside the disk D_i, given a drawing Γ_i of G_i, and let $\mathcal{D}(\Gamma_i)$ be the set of all such drawings. L- and C-drawings represent the tracks as straight lines. Therefore, $\Delta(\Gamma_i)$ is only relevant for O-drawings and we leave it undefined for the other two drawing styles. For this drawing style, we can again represent $\Delta(\Gamma_i)$ combinatorially, ensuring that $\mathcal{D}(\Gamma_i)$ is finite. We define $\mathcal{G}_{\max} \coloneqq \max_{i\in[k]} |\mathcal{G}_i|$ and $\mathcal{D}_{\max} \coloneqq \max_{\Gamma_i\in\mathcal{G}_i, i\in[k]} |\mathcal{D}(\Gamma_i)|$ as the maximum number of drawings of $G_1, \ldots, G_k$ and their corresponding track drawings, respectively.

In our algorithm, we maintain an $\mathcal{O}(k \cdot \mathcal{G}_{\max} \cdot \mathcal{D}_{\max})$-size table C, where $C[i, \Gamma_i, \Delta(\Gamma_i)]$ stores the minimum number of crossings of a drawing for the bags $V_1, V_2, \ldots, V_i$, where Γ_i is the drawing of G_i in D_i and $\Delta(\Gamma_i)$ the drawing of its incident tracks. We call $S_i = (i, \Gamma_i, \Delta(\Gamma_i))$ a *state* of our DP. To compute $C[S_i]$, we need to count all e/e-crossings in Γ_i involving edges from E_i, and all t/e-, and t/t-crossings involving also edges or tracks incident to vertices from V_{i-1} for a given state S_{i-1}, if the bag V_{i-1} exists. Let $\mathrm{cr}_{\mathsf{e/e}}(\cdot)$, $\mathrm{cr}_{\mathsf{t/e}}(\cdot,\cdot)$, and $\mathrm{cr}_{\mathsf{t/t}}(\cdot,\cdot)$ denote these number of crossings, respectively. Note that the exact definition of $\mathrm{cr}_{\mathsf{e/e}}(\cdot)$, $\mathrm{cr}_{\mathsf{t/e}}(\cdot,\cdot)$, and $\mathrm{cr}_{\mathsf{t/t}}(\cdot,\cdot)$ depends on the desired drawing style. For $i > 1$, the number of crossings $\mathrm{cr}(S_i, S_{i-1})$ for two states $S_i = (i, \Gamma_i, \Delta(\Gamma_i))$ and $S_{i-1} = (i-1, \Gamma_{i-1}, \Delta(\Gamma_{i-1}))$ equals the sum of $\mathrm{cr}_{\mathsf{e/e}}(S_i)$, $\mathrm{cr}_{\mathsf{t/e}}(S_i, S_{i-1})$, and $\mathrm{cr}_{\mathsf{t/t}}(S_i, S_{i-1})$; note that $\mathrm{cr}(S_i, S_{i-1})$ can also be defined as a weighted sum to prioritize some crossings over others; recall Section 3. We can now relate the different states in C to each other as follows, where $S_i = (i, \Gamma_i, \Delta(\Gamma_i))$, $i \in [k]$, $\Gamma_i \in \mathcal{G}_i$, and $\Delta(\Gamma_i) \in \mathcal{D}(\Gamma_i)$:

$$C[S_i] = \begin{cases} \mathrm{cr}_{\mathsf{e/e}}(S_i) & \text{if } i = 1 \\ \min\limits_{\substack{S_{i-1} = (i-1,\Gamma_{i-1},\Delta(\Gamma_{i-1})),\\ \Gamma_{i-1}\in\mathcal{G}_{i-1},\ \Delta(\Gamma_{i-1})\in\mathcal{D}(\Gamma_{i-1})}} C[S_{i-1}] + \mathrm{cr}(S_i, S_{i-1}) & \text{otherwise} \end{cases} \tag{1}$$

Using induction, we can show that Equation (1) captures the minimum number of crossings.

Lemma 1. *Let T be a path decomposition of G with k bags. For each $i \in [k]$, drawings $\Gamma_i \in \mathcal{G}_i$ and $\Delta(\Gamma_i) \in \mathcal{D}(\Gamma_i)$, the table entry $C[i, \Gamma_i, \Delta(\Gamma_i)]$ equals the minimum number of crossings of a witness drawing for the bags $V_1, \ldots, V_i$ with the drawings Γ_i and $\Delta(\Gamma_i)$ for G_i and its incident tracks, respectively.*

Proof: We use induction over i to show correctness. Throughout the proof, we let Γ_i and $\Delta(\Gamma_i)$ be two arbitrary drawings.

Base Case ($i = 1$). For $i = 1$, we are seeking a drawing of the bag V_1 only, i.e., a drawing of G_1 in the disk D_1. Clearly, the minimum number of crossings equals the number of e/e-crossings in the disk D_1, which is precisely $\mathrm{cr}_{\mathsf{e}/\mathsf{e}}(1, \Gamma_1, \Delta(\Gamma_1))$; see also the first case in Equation (1). Thus, $C[1, \Gamma_1, \Delta(\Gamma_1)]$ correctly captures the required crossing count.

Inductive Step. Let $i > 1$ and note that we are in the second case of Equation (1). Furthermore, let $C[i, \Gamma_i, \Delta(\Gamma_i)] = c$ and assume, for the sake of a contradiction, that there exists a crossing-minimal witness drawing Γ^* for the first i bags and with the drawings Γ_i and $\Delta(\Gamma_i)$ for the disk D_i. Let the number of crossings for Γ^* be some $c^* < c$. We now consider the drawing Γ' on the first $i-1$ bags that is induced by Γ^* and denote by c' its number of crossings. Let Γ_{i-1} be the drawing of G_{i-1} and $\Delta(\Gamma_{i-1})$ be the drawing of its incident tracks in Γ'. Clearly, c' is the minimum number of crossings of a witness drawing for the first $i-1$ bags of T that uses the drawings Γ_{i-1} and $\Delta(\Gamma_{i-1})$ for G_{i-1} and its incident tracks, respectively. If this would not be the case, then Γ^* would not be crossing-minimal. By our inductive hypothesis, we have $C[i-1, \Gamma_{i-1}, \Delta(\Gamma_{i-1})] = c'$. Note that we consider in the second case of Equation (1) all possible states for $i-1$, and in particular the state $(i-1, \Gamma_{i-1}, \Delta(\Gamma_{i-1}))$. Since we take among all such states the one that leads to a minimum number of crossings, the fact that we have $C[i, \Gamma_i, \Delta(\Gamma_i)] = c > c^*$ yields a contradiction to the existence of Γ^* with c^* crossings, as we would have set $C[i, \Gamma_i, \Delta(\Gamma_i)]$ to its number of crossings c^* otherwise. Hence, by induction, the lemma statement follows. □

With Lemma 1 at hand, we now have all key ingredients for our DP. It remains to define for each drawing style the representations of Γ_i and $\Delta(\Gamma_i)$, if needed, and to implement the crossing counting functions $\mathrm{cr}_{\mathsf{e}/\mathsf{e}}(\cdot)$, $\mathrm{cr}_{\mathsf{t}/\mathsf{e}}(\cdot,\cdot)$, and $\mathrm{cr}_{\mathsf{t}/\mathsf{t}}(\cdot,\cdot)$. This will be the main task for the remainder of this and the upcoming sections.

Two-Page Linear Drawings. We now aim to compute a crossing-minimal $\mathsf{L2}$-drawing of a path decomposition T. Recall that a two-page book drawing Γ_i of $G_i = (V_i, E_i)$ is uniquely defined by the linear order $\prec_i$ on V_i and the page assignment $\sigma_i \colon E_i \to \{\ell, r\}$. Thus, we set $\Gamma_i = \langle \prec_i, \sigma_i \rangle$. As the tracks are drawn using straight lines between the vertices in the respective disks, it is sufficient to store only the drawing for each G_i, $i \in [k]$, in our DP table C. As $\mathcal{G}_{\max} = \mathcal{O}(\overline{\omega}! \cdot 2^{\overline{\omega}^2})$ holds, the size of the table C is in $\mathcal{O}(k \cdot \overline{\omega}! \cdot 2^{\overline{\omega}^2})$ and it only remains to implement the crossing counting functions. To this end, recall that we restrict ourselves to combinatorial crossings.

For e/e-crossings, we observe that $\mathrm{cr}_{\mathsf{e}/\mathsf{e}}(\cdot)$ equals the number of edges with alternating endpoints in $\prec_i$ but assigned in σ_i to the same page; see Figure 6a and Equation (2).

$$\mathrm{cr}_{\mathsf{e}/\mathsf{e}}(S_i) \coloneqq |\{(u,v),(a,b) \in E_i \mid \sigma((u,v)) = \sigma((a,b)), u \prec_i a \prec_i v \prec_i b\}| \tag{2}$$

To determine the number of t/e-crossings, it is sufficient to observe that an edge $(u,v) \in E_{i-1}$ with $u \prec_{i-1} v$ crosses with a track for the vertex $w \in V_{i-1} \cap V_i$ if and only if $u \prec_{i-1} w \prec_{i-1} v$ and $\sigma_{i-1}((u,v)) = r$ as visualized in Figure 6b. A symmetric observation can be made for edges $(u,v) \in E_i$ with $\sigma_i((u,v)) = \ell$, yielding the following function:

$$\begin{aligned} \mathrm{cr}_{\mathsf{t}/\mathsf{e}}(S_i, S_{i-1}) \coloneqq & \sum_{\substack{(u,v)\in E_{i-1} \\ \sigma_{i-1}((u,v))=r}} |\{w \in V_{i-1} \cap V_i \mid u \prec_{i-1} w \prec_{i-1} v\}| \\ & + \sum_{\substack{(u,v)\in E_i \\ \sigma_i((u,v))=\ell}} |\{w \in V_{i-1} \cap V_i \mid u \prec_i w \prec_i v\}| \end{aligned} \tag{3}$$

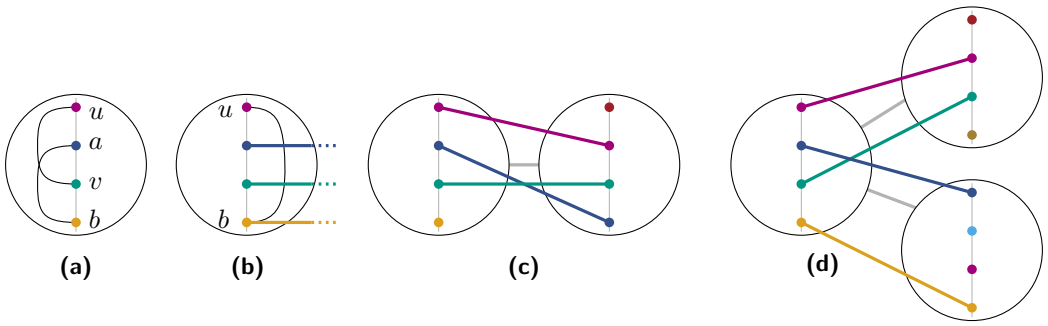


Figure 6: Illustration of different types of crossing: **(a)** The e/e-crossing between the two edges (u, v) and (a, b) is enforced by the relative order of their endpoints. **(b)** The edge (u, b) crosses the blue and green tracks because the respective vertices lie between u and b. **(c)** The blue and green tracks cause a t/t-crossing because the relative order among the respective endpoints is inversed. **(d)** A "criss-cross" t/t-crossing between blue and green that can occur in tree decompositions.

This leaves us with counting the t/t-crossings, for which it is sufficient to count the number of inversions between $\prec_{i-1}$ and $\prec_i$; see Figure 6c and Equation (4).

$$\mathrm{cr}_{\mathsf{t/t}}(S_i, S_{i-1}) := |\{u, v \in V_{i-1} \cap V_i \mid u \prec_{i-1} v, v \prec_i u\}| \tag{4}$$

Combining this with the recurrence relation from Equation (1), we obtain the following:

Theorem 2. *Let T be a path decomposition of G of width ω. We can compute a crossing-minimal L2-drawing of T in $\mathcal{O}(n \cdot (\overline{\omega}!)^2 \cdot 4^{\overline{\omega}^2} \cdot \overline{\omega}^4)$ time.*

Proof: We employ our DP algorithm storing the above-described states to compute a crossing-minimal L2-drawing of T. Recall that the table C has size $\mathcal{O}(k \cdot \overline{\omega}! \cdot 2^{\overline{\omega}^2})$, where $k = \mathcal{O}(n)$ denotes the number of bags in T. To compute a single state for some $i \in [k]$, we have to access up to $\mathcal{O}(\overline{\omega}! \cdot 2^{\overline{\omega}^2})$ states for $i - 1$. A closer analysis of Equations (2) to (4) reveals that we can evaluate the function $\mathrm{cr}(\cdot, \cdot)$ in $\mathcal{O}(\overline{\omega}^4)$ time. Thus, we can compute the minimum number of crossings for a given state in $\mathcal{O}(\overline{\omega}! \cdot 2^{\overline{\omega}^2} \cdot \overline{\omega}^4)$ time. Overall, this amounts to $\mathcal{O}(k \cdot (\overline{\omega}!)^2 \cdot 4^{\overline{\omega}^2} \cdot \overline{\omega}^4)$ time to fill the entire table C. The minimum number of crossings can be obtained by taking the minimum over all $C[k, \cdot, \cdot]$. We can use standard backtracking techniques to compute the crossing-minimal L2-drawing of T. The correctness of our algorithm follows directly from the correctness of our recurrence relation, i.e., Equation (1), established in Lemma 1. □

By restricting the DP table C and relation from Equation (1), we can generalize the above-presented DP. For example, to compute an L1-drawing, we can remove the page assignment from our definition of a drawing Γ_i of G_i. Furthermore, it could be desired to enforce a consistent linear order across different bags, i.e., forbid t/t-crossings. To that end, we can consider in Equation (1) only those linear orders $\prec'$ that are consistent with $\prec$, i.e., where $\prec|_{V_{i-1}} = \prec'|_{V_i}$. Below, we summarize the effects on the running time.

Corollary 1. *Let T be a path decomposition of G of width ω. We can compute a crossing-minimal*

- *L1-drawing of T in $\mathcal{O}(n \cdot (\overline{\omega}!)^2 \cdot \overline{\omega}^4)$ time.*
- *L2-drawing of T without t/t-crossings in $\mathcal{O}(n \cdot (\overline{\omega}!)^2 \cdot 4^{\overline{\omega}^2} \cdot \overline{\omega}^4)$ time.*

5.2 Linear Drawings for Tree Decompositions

Similarly, for graphs of bounded treewidth, we can obtain the minimum number of total crossings for L-drawings via DP. Given a tree decomposition T of G, we orient T "left-to-right", where the root bag is leftmost and children are to the right of the parent. We then obtain a right-to-left ordering of the tree's bags, $V_k, V_{k-1}, \ldots, V_1$, where V_1 is the root bag.

DP for Treewidth. Our DP formulation in this problem uses the same table as for path decompositions, but a given bag may now have more than one previous bag that can affect its crossings. We assume that the tree decomposition T only has bags of degree up to three. As such, any bag has at most one parent (due to the "left-to-right" orientation) and at most two children. We consider three cases to calculate $C[S_i]$ for a bag V_i, based on its in-degree $\deg^-(V_i) \in \{0, 1, 2\}$. The cases where $\deg^-(V_i)$ equal zero or one are similar to the two cases presented in the DP for pathwidth, but when $\deg^-(V_i) = 2$ the crossing function $\mathrm{cr}(\cdot,\cdot,\cdot)$ depends on the current state and its two child states. In this case, the current bag has two children belonging to its in-degree neighboring set $N^-(V_i)$ whose embedding (which child is above the other) is decided by the DP. Our DP table has the following recurrence relation:

$$C[S_i] = \begin{cases} \mathrm{cr}_{\mathsf{e}/\mathsf{e}}(S_i) & \text{if } \deg^-(V_i) = 0 \\ \min\limits_{\substack{S_x \,=\, (x,\Gamma_x,\Delta(\Gamma_x)) \\ V_x \in N^-(V_i)}} C[S_x] + \mathrm{cr}(S_i, S_x) & \text{if } \deg^-(V_i) = 1 \\ \min\limits_{\substack{(S_x,S_y) \\ V_x,V_y \in N^-(V_i), x \neq y}} C[S_x] + C[S_y] + \mathrm{cr}(S_i, S_x, S_y) & \text{if } \deg^-(V_i) = 2 \end{cases} \tag{5}$$

Two-Page Linear Drawings. The number of e/e-crossings does not change within the parent bag, so that remains the same as in Equation (2). The number of t/e-crossings can be treated independently for both children $V_x, V_y \in N^-(V_i)$, as there are no such crossings between the children. So Equation (6) is similar to Equation (3) except that we sum over both children.

$$\mathrm{cr}_{\mathsf{t}/\mathsf{e}}(S_i, S_x, S_y) \coloneqq \sum_{i' \in \{x,y\}} \left(\sum_{\substack{(u,v) \in E_{i'} \\ \sigma_{i'}((u,v)) = \ell}} |\{w \in V_{i'} \cap V_i \mid u \prec_{i'} w \prec_{i'} v\}| \right.$$
$$\left. + \sum_{\substack{(u,v) \in E_i \\ \sigma_i((u,v)) = r}} |\{w \in V_{i'} \cap V_i \mid u \prec_i w \prec_i v\}| \right) \tag{6}$$

At this point, we want to make the reader aware that we have to maintain a sufficient x- and y-distance of the bags to each other (or, alternatively, a correct radius for the arcs) to ensure that we only encounter combinatorial crossings in the geometric drawing. Otherwise, for example, additional t/e-crossings can occur between edges and tracks incident to the same vertex. We note, however, that there is always a drawing containing precisely the determined combinatorial crossings, although with no guarantee on its resolution.

For t/t-crossings, we must handle t/t-crossings between children. Without loss of generality, let the child V_x that comes first in the input be above child V_y. Then such t/t-crossings will occur when two tracks from the children "criss-cross", where a track from the top child goes below the track from the bottom child as in Figure 6d, captured by Equation (7). Then we add the t/t-crossings between parent and child as in Equation (4), but for both children now:

$$\mathrm{cr}_{\mathsf{t}/\mathsf{t}}(S_i, S_x, S_y) \coloneqq |\{(u,v) \in (V_x \cap V_i) \times (V_y \cap V_i) \mid v \prec_i u\}| \tag{7}$$

$$+ \sum_{i' \in \{x,y\}} |\{u, v \in V_{i'} \cap V_i \mid u \prec_{i'} v, v \prec_i u\}| \tag{8}$$

A similar analysis to Theorem 2 and Corollary 1 yields the following results for L-drawings:

Theorem 3. *Let T be a tree decomposition of G of width ω. We can compute a crossing-minimal L2-drawing of T in $\mathcal{O}(n \cdot (\overline{\omega}!)^3 \cdot 8^{\overline{\omega}^2} \cdot \overline{\omega}^4)$ time.*

Proof: The table C has the same size $\mathcal{O}(k \cdot \overline{\omega}! \cdot 2^{\overline{\omega}^2})$, with $k = \mathcal{O}(n)$, and computing $\mathrm{cr}(\cdot,\cdot,\cdot)$ still takes $\mathcal{O}(\overline{\omega}^4)$ time as in the pathwidth case. However, computing a single state must now check all possible pairings of its two child states, yielding $\mathcal{O}((\overline{\omega}! \cdot 2^{\overline{\omega}^2})^2)$ states in the worst case. So, altogether, the running time will be in $\mathcal{O}(n \cdot (\overline{\omega}!)^3 \cdot 8^{\overline{\omega}^2} \cdot \overline{\omega}^4)$. □

Corollary 2. *Let T be a tree decomposition of G of width ω. We can compute a crossing-minimal L1-drawing of T in $\mathcal{O}(n \cdot (\overline{\omega}!)^3 \cdot \overline{\omega}^4)$.*

Finally, we note that our DP algorithm can be extended to decompositions T with bags of arbitrary degree by generalizing Equations (5) to (7). In particular, for Equation (7), we must account for t/t-crossings between every pair of children of V_i. As stated in Equation (5), we consider all possible combinations of states for V_i's children, and the number of such combinations grows with their number. Moreover, observe that the embedding of these children affects the number of t/t-crossings, and the number of potential embeddings is itself exponential in the degree $\deg(V_i)$ of V_i in T. Therefore, allowing bags with non-constant degree in T introduces an exponential factor in terms of the maximum degree of T in the running time of our DP algorithm.

5.3 Drawing Styles with Circular Vertex Placements

We now discuss how to adapt the above-presented DP algorithms to compute crossing-minimal circular (C-) and orbital (O-) drawings. Recall that for C- and O-drawings, the angle parameter α specifies the starting angle of the first vertex in each disk; recall also Figure 4. Since finding a good value of α is an interesting problem in its own right, we consider, in the following, α to be fixed and highlight the steps that depend on the concrete value of α.

Circular Drawings. Circular drawings arrange the vertices in every disk D_i, $i \in [k]$, on a circle and draw the edges E_i as straight lines. Thus, the drawing Γ_i of G_i is uniquely defined by the placement of the vertices V_i in D_i. Since they are evenly distributed on a circle, it suffices to store in Γ_i for each bag $V_i \in T$, the counterclockwise order $\prec_i$ of the vertices V_i as they appear in D_i, starting at the twelve o'clock position, where the angle parameter α specifies the starting angle of the first vertex in the order $\prec_i$ in the disk D_i. As the tracks are again straight lines, we do not need to store $\Delta(\Gamma_i)$, reducing the size of C to $\mathcal{O}(k \cdot \overline{\omega}!)$.

When filling the table C, we recall that the recurrence relations from Equations (1) and (5) include the number of e/e-, t/e-, and t/t-crossings involving the vertices of a bag V_i, $i \in [k]$, and those of its adjacent bags (if they exist). The number of e/e-crossings corresponds, due to the equivalence with one-page book drawings, to the number of edge pairs with alternating endpoints in $\prec_i$ and is, therefore, purely combinatorial in this setting. In contrast, the t/e-, and t/t-crossings are geometric in nature and depend on the concrete positions of the vertices within the disks.

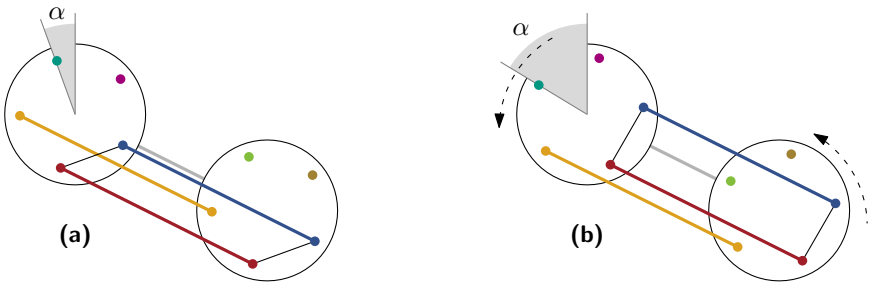


Figure 7: The observed crossings not only depend on the order of the vertices in the disks but also on the choice of α (also indicated with the dashed arrows): The t/e-crossing with the yellow track in **(a)** is not present in **(b)** although we use the same linear orders in both visualizations.

Observe that these positions depend on α and we remark that the choice of α influences the number of observed crossings; see also Figure 7. Consequently, we equip the crossing counting functions $\mathrm{cr}_{\mathsf{t/e}}(\cdot,\cdot,\cdot)$ and $\mathrm{cr}_{\mathsf{t/t}}(\cdot,\cdot,\cdot)$ with the parameter α. Altogether, this is sufficient to compute crossing-minimal C-drawings.

Orbital Drawings. Orbital drawings extend circular drawings by routing the tracks as orbits around the vertices; recall Figure 2d. Since tracks must remain within the track-routing area, every O-drawing is free of t/e-crossings. However, unlike in the other two drawing styles, the drawing of tracks within a disk is no longer uniquely determined by the placement of the vertices. In particular, tracks can be assigned to different orbits and may orbit the drawing Γ_i of G_i either clockwise or counterclockwise. Thus, while it still suffices to store a linear order $\prec_i$ of V_i for Γ_i, representing $\Delta(\Gamma_i)$ now requires additional care.

We model $\Delta(\Gamma_i)$ with two parts. First, to model the assignment of tracks to orbits, we introduce an *orbit assignment function* $\lambda_i\colon V_i \to [\overline{\omega}]$, where $\lambda_i(v)$ specifies the orbit used for the tracks of vertex v, numbered from the center of the disk D_i outward; see Figure 8. Two tracks for different vertices $u, v \in V_i$ may share the same orbit if they do not intersect. Otherwise, we require $\lambda_i(u) \neq \lambda_i(v)$ to guarantee that no two tracks for different vertices share the same orbit simultaneously. Second, to model the direction in which the tracks orbit, i.e., clockwise or counterclockwise, we consider the (up to two) children V_x and V_y and the parent V_q of the bag V_i in the tree decomposition T, if they exist. We define a *direction function* $\delta_i\colon V_i \times \{x, y, q\} \to \{\mathrm{cw}, \mathrm{ccw}\}$ which, for each vertex $v \in V_i$, captures the direction (clockwise (cw) or counterclockwise (ccw)) of the track from V_i to each adjacent bag. Figure 8 illustrates the semantic of δ_i and underlines the importance of storing this information separately for each adjacent bag. Thus, for a given drawing Γ_i of G_i, we set $\Delta(\Gamma_i) = (\lambda_i, \delta_i)$. This increases the size of the DP table C to $\mathcal{O}(k \cdot \overline{\omega}! \cdot (8\overline{\omega})^{\overline{\omega}})$.

Regarding the computation of the number of crossings, we observe that the function $\mathrm{cr}_{\mathsf{e/e}}(\cdot)$ from C-drawings can be reused, and that t/e-crossings do not occur in this drawing style. t/t-crossings involving a track for a vertex $v \in V_i$ occur in the following two cases. First, if v also appears in an adjacent bag, and a track for some vertex $u \in V_i$ with $\lambda_i(u) < \lambda_i(v)$ passes by the position of v in Γ_i, then these two tracks cross: The track for v must cross the one of u to reach its orbit; see Figure 9a. Second, for each adjacent bag V_j with $v \in V_j$, we count the number of vertices $u \in V_i$ such that a track for u blocks the path from D_i to D_j, see Figure 9b. In both cases, the crossings can be determined from the starting angle α, the order of the children, and the stored information, i.e., Γ_i and $\Delta(\Gamma_i)$, for the disk D_i of V_i and the disks of its adjacent bags. Hence, the

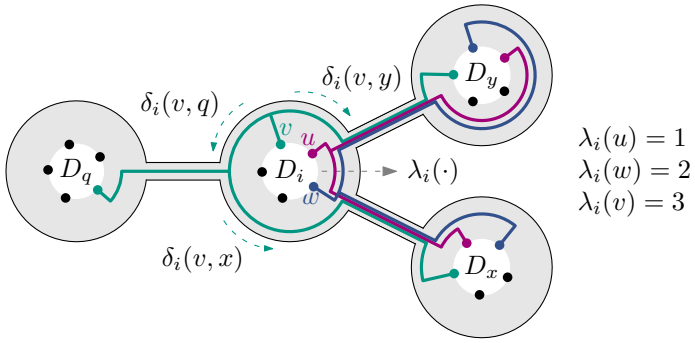


Figure 8: Visualization of a (partial) O-drawing and the information that we store in the table C. The track routing area is indicated in gray. Black vertices represent arbitrary other vertices. In this drawing, we have $\lambda_i(u) = 1$, $\lambda_i(w) = 2$, $\lambda_i(v) = 3$, $\delta_i(v,y) = \text{cw}$, and $\delta_i(v,q) = \delta_i(v,x) = \text{ccw}$. Note that these orientations are required to minimize the t/t-crossings involving the tracks for vertex v.

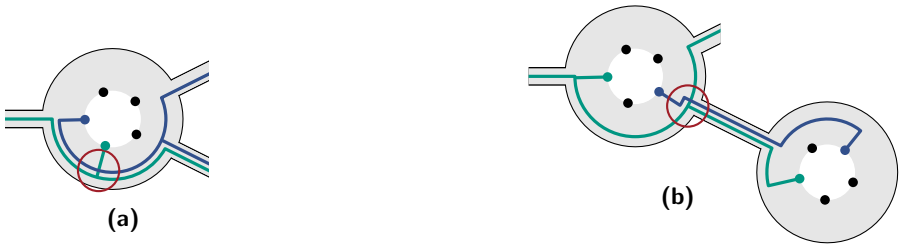


Figure 9: The two cases how t/t-crossings in O-drawings can arise: In **(a)**, the blue tracks orbit inside the green ones, thus causing a crossing when they pass the green vertex. t/t-crossings are highlighted with the red circles. The crossing in **(b)** occurs because the green track blocks the visibility to the lower-right disk for the blue track. Observe that there is only one t/t-crossing in **(a)**, although there is technically one blue track from the upper- and one from the lower-right disk.

function $\text{cr}_{\text{t/t}}(\cdot,\cdot,\cdot)$ still only depends on the states for V_i and its child bags. Finally, since tracks to occurrences of the same vertex $v \in V_i$ in different bags can overlap, we must avoid double-counting in such cases; see again Figure 9a. We summarize below the findings of this section. Recall that $k = \mathcal{O}(n)$ holds.

Theorem 4. *Let T be a decomposition of G of width ω. We can compute a crossing-minimal*

- *C-drawing of T in $\mathcal{O}(n \cdot (\overline{\omega}!)^2 \cdot \overline{\omega}^4)$ time if T is a path decomposition.*
- *C-drawing of T in $\mathcal{O}(n \cdot (\overline{\omega}!)^3 \cdot \overline{\omega}^4)$ time if T is a tree decomposition.*
- *O-drawing of T in $\mathcal{O}(n \cdot (\overline{\omega}!)^2 \cdot (4\overline{\omega})^{2\overline{\omega}} \cdot \overline{\omega}^4)$ time if T is a path decomposition.*
- *O-drawing of T in $\mathcal{O}(n \cdot (\overline{\omega}!)^3 \cdot (8\overline{\omega})^{3\overline{\omega}} \cdot \overline{\omega}^4)$ time if T is a tree decomposition.*

Proof: Let k denote the number of bags of T. Recall $k = \mathcal{O}(n)$ and that our dynamic program (DP) traverses the decomposition T and considers every bag $V_i \in T$, and all the possible drawings Γ_i and $\Delta(\Gamma_i)$ of G_i and its incident tracks, respectively. For each of them, all possible combinations with states of the two children of V_i, if they exist, or V_{i-1}, in case T is a path decomposition. Lemma 1 and the arguments in Sections 5.1 and 5.2 ensure correctness of this DP for our definitions of states in the DP. Therefore, we focus for the remainder of the proof on the claimed running times and discuss C- and O-drawings separately.

C-drawings. Recall that a drawing Γ_i for G_i, $i \in [k]$ can, thanks to the angle parameter α, be defined by the counterclockwise order $\prec_i$ of the vertices V_i starting at the twelve o'clock position. As the tracks are straight lines, the size of the DP-table C is $\mathcal{O}(k \cdot \overline{\omega}!)$. The position for each vertex can be computed in $\mathcal{O}(1)$ time. Thus, the evaluation of the crossing counting function $\mathrm{cr}(\cdot,\cdot)$ still requires $\mathcal{O}(\overline{\omega}^4)$ time as for L-drawings. For computing one state of the DP, we must check all $\mathcal{O}((\overline{\omega}!)^2)$ combinations of the up to two children states. This yields the claimed running for tree decompositions. For path decompositions, only $\mathcal{O}(\overline{\omega}!)$ states need to be evaluated as each bag has degree at most two in T.

O-drawings. Recall that for O-drawings, we need to specify in addition the drawing of the tracks inside the disk D_i, $i \in [k]$, using an orbit assignment function λ_i and a direction function δ_i. As $\mathcal{G}_{\max} = \mathcal{O}(\overline{\omega}!)$ and $\mathcal{D}_{\max} = \mathcal{O}(\overline{\omega}^{\overline{\omega}} \cdot 2^{3\overline{\omega}})$, this results in a DP-table C of size $\mathcal{O}(k \cdot \overline{\omega}! \cdot (8\overline{\omega})^{\overline{\omega}})$. As for C-drawings, we can compute the position for each vertex in $\mathcal{O}(1)$ time, which yields the drawing Γ_i of G_i. Together with $\Delta(\Gamma_i) = (\lambda_i, \delta_i)$, this is sufficient to describe the drawing of the tracks in the disk D_i. Despite not having t/e-crossings, computing the number $\mathrm{cr}(\cdot,\cdot)$ of crossing still takes $\mathcal{O}(\overline{\omega}^4)$ time. Since we have to evaluate for each state all combinations with the $\mathcal{O}(\overline{\omega}! \cdot (8\overline{\omega})^{\overline{\omega}})$-many states for each of its up to two children, the claimed running time for tree decompositions follows. If T is a path decomposition, the fact that T has degree at most two reduces the size of the table C to $\mathcal{O}(k \cdot \overline{\omega}! \cdot (4\overline{\omega})^{\overline{\omega}})$ and the evaluation time for a single state to $\mathcal{O}((\overline{\omega}!) \cdot (4\overline{\omega})^{\overline{\omega}} \cdot \overline{\omega}^4)$. □

6 Heuristic Approaches for Linear Witness Drawings

A proof-of-concept implementation of our DP from Section 5.2 for L2-drawings of tree decompositions required over 8 minutes to terminate even for small decompositions of width $\omega = 3$ with four bags. This is no surprise given the running time bounds established in Theorem 3. For tree decompositions of larger width, this became too large in practice with a single state needing to check up to $(5! \cdot 2^{5^2})^2 \approx 10^{19}$ possible configurations for $\omega = 4$.

Consequently, we do not want to restrict our attention to exact, but slow, algorithms, but also explore heuristics to efficiently compute drawings with a small number of crossings, but without any formal guarantee on optimality. In this section, we describe three heuristics for L2-drawings. In Section 7, we will evaluate their performance and provide sample drawings computed by them.

Global Book Drawing Heuristic. Since we visualize in an L2-drawing each graph G_i, $i \in k$, as a two-page book drawing of G_i, our first heuristic computes such a drawing $\langle \prec, \sigma \rangle$ for the entire graph G. After that, we set for each disk D_i $\prec_i = \prec|_{V_i}$ and $\sigma_i(e) = \sigma(e)$ for every edge $e \in E_i$, i.e., we project the drawing $\langle \prec, \sigma \rangle$ down into each bag. If T is a tree decomposition, we additionally perform a bottom-up traversal of T, choosing for each internal bag V_i the embedding of its children that yields the fewer t/e- and t/t-crossings locally.

Since computing a crossing-minimal one-page book drawing for a graph is an NP-hard problem [55], we already employ a heuristic for this step. Klawitter, Mchedlidze, and Nöllenburg [47] performed an extensive experimentally evaluation on book drawing heuristics. In their conclusion, they recommended the heuristic `conGreedy+`, which places one vertex after the other on the spine, selecting the next vertex based on the number of already placed neighbors, and extends the spine order and page assignment greedily. While for two-page book drawings of k-trees, which are maximal graphs of treewidth k, it was slightly outperformed by other heuristics, it performed best overall and in particular for graphs with a sub-quadratic number of edges. The heuristic has a running time of $\mathcal{O}(m^2 n)$ [47] and we call it `Global conGreedy+`.

Local Book Drawing Heuristic. Computing a book drawing for the whole graph G and then applying it to each disk D_i, $i \in k$, has two main disadvantages. On the one hand, we usually do not visualize the entire graph G in a single bag, but several induced subgraphs G_i separately. Therefore, the drawing $\langle \prec_i, \sigma_i \rangle$ for G_i might contain unnecessary crossings. On the other hand, the above-described heuristic is, apart from the embedding step, unaware of t/e- and t/t-crossings. This, in particular, leads to situations where swapping the page assignment of one edge $e \in E_i$ or the entire edge set E_i can dramatically reduce the number of t/e-crossings. The latter occurs especially at the root and leave bags of T.

Therefore, we now employ the algorithm `conGreedy+` in each bag V_i separately during a top-down traversal of T, thus computing for each G_i a book drawing from scratch. Since the drawing for the parent of a bag V_i has been determined, we can take the potential t/t- and t/e-crossings into account. More precisely, when selecting the spine position of a vertex $v \in V_i$ and the page assignment for its incident edges in G_i we also compute the number of t/e- and t/t-crossings that this causes based on the drawing of the parent, similar to Equations (6) and (7). Furthermore, we forward this information to its children when considering their two possible embeddings if necessary. We call this heuristic `Local conGreedy+` and, since the size of each G_i is bounded, its running time is $\mathcal{O}(n \cdot \overline{\omega}^5)$.

Local Search. In addition to the two above-mentioned heuristics, we also perform a local search to reduce the number of crossings. Given an L2-drawing of T, we traverse T bottom up. For each bag $V_i \in T$, we perform one or several of the following movements.

Vertex-Swap: Take two vertices $u, v \in V_i$ and swap their position on the spine $\prec_i$.

Edge-Swap: Take two edges $e_1, e_2 \in E_i$ with $\sigma_i(e_1) \neq \sigma_i(e_2)$ and swap their page assignment.

Edge-Flip: Take an edge $e \in E_i$ and assign it to the other page.

Embedding-Flip: Flip the embedding of the subtree V_i, i.e., the order of its children.

For a given bag $V_i \in T$, we perform a hill-climbing approach and exhaustively apply above movements until no further improvement can be made in the drawing $\langle \prec_i, \sigma_i \rangle$. Then, we proceed to the next bag $V_j \in T$. Note that when evaluating the number of crossings, we keep the drawings in the remaining bags fixed. Thus, after a bottom-up traversal of T, we perform a second traversal, this time top-down. Afterwards, or after a predetermined period of time, we return the best solution found so far.

7 Experimental Evaluation

We implemented the dynamic program for L2-crossings as described in Section 5.2 as well as all of the heuristics presented in Section 6 in `Python 3.6.9`. All algorithms run on a system with Intel Xeon E5-2640 v4 10-core processors at 2.40 GHz. We set hard limits for memory and time of 96 GB and fifteen minutes, respectively, but memory was not a limitation for our algorithms. In particular, even on the largest instances, our dynamic program used less than 2 GB of memory on average. If we improve a solution from a heuristic using local search, we indicate this using "`& LS`". Also the local search has a maximum running time of fifteen minutes. The computed drawings are visualized using `d3.js` on an independent system and this final step is not considered here. The instances are based on a public repository containing graphs extracted from "Sage" together with a tree decomposition of them.[2] Out of the 150 graphs, 55 of them were equipped with a (tree or path) decomposition T where each bag, i.e., node of the decomposition T, has degree at most three in T and we used these in our experiments. We have no information on how these decompositions were computed or if they are optimal. The width of the decompositions ranges between one and 53 and they have between one and 143 bags. For 60% of the instances, both parameters were simultaneously below ten. The graphs have between four and 143 vertices and between six and 1,008 edges, with a mean density, i.e., $2m/(n(n-1))$, of 0.37.

Computed Drawings. We provide in Figures 10 and 11 sample drawings of two graphs together with the witness drawings computed by our different algorithms. Generally, we can see that the DP and `Global conGreedy+` produce drawings with a more consistent arrangement of the vertices on the individual spines. Comparing Figures 10b and f, we can observe that the obtained drawing from `Global conGreedy+ & LS` is nearly identical to the one of the DP, confirming the observed low crossing numbers. The consistency across different spines yields also a visually more pleasing drawing, as we can see in Figure 11. We provide in Appendix A more sample drawings computed by our algorithms and analyze below their running time and the quality of the produced drawings.

Running Times, Tradeoffs, and the Influence of the Width. All heuristics terminated on every instance within fifteen minutes. The DP could solve only thirteen instances within the time limit. To evaluate the crossing minimization performance of the heuristics on a larger set of instances, we ran the DP with a time limit of six hours, after which 21 of the 55 instances, all of width at most four, could be solved optimally. For the remaining 34 instances, `Global conGreedy+` obtained four, `Local conGreedy+` three, `Global conGreedy+ & LS` 18, and `Local conGreedy+ & LS` 20 times a solution with the fewest crossings.

Figure 12 shows the relative number of crossings, compared to the best-performing algorithm, i.e., the DP where available and one of the heuristics otherwise, and running time for each algorithm and instance. We group instances by width and sort them within each group based on the number of bags. In Figure 13, we provide a filtered version of Figure 12, focusing on instances for which the DP terminated within six hours. Regarding the running time, we can observe in Figure 12a that all heuristics without the local search step terminated within a second on every instance except the largest ones. As expected, `Global conGreedy+` is the fasted heuristic, since it has to compute only one book drawing for the entire graph. Regarding the optimality-ratio, Figure 12b shows that

[2] The original dataset is available at https://github.com/freetdi/named-graphs. The filtered dataset, source code, and evaluation code can be found on OSF [17]. The (interactive) application can also be accessed online at https://www.ac.tuwien.ac.at/projects/visualizing-treewidth/.

[4] A drawing of the graph can be found online: https://en.wikipedia.org/wiki/Brinkmann_graph.

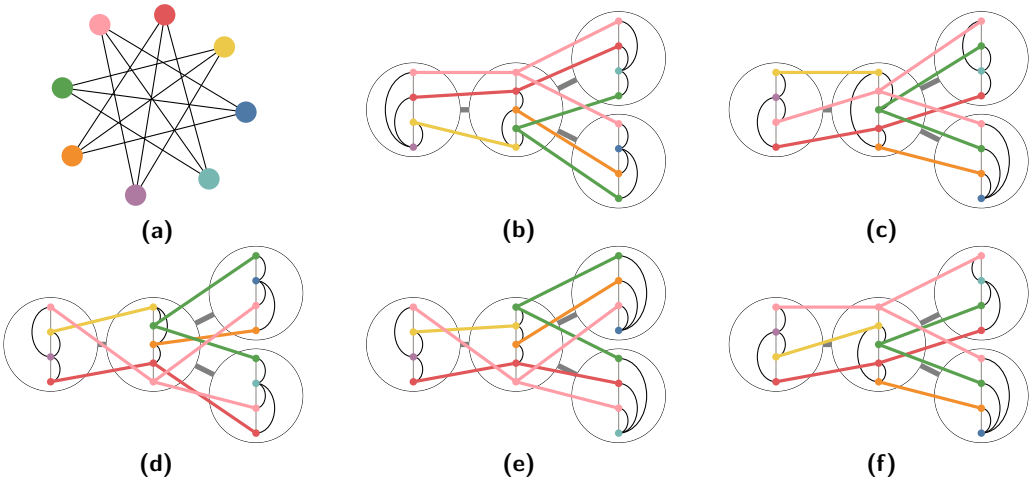


Figure 10: Comparison of the drawings computed by our algorithms for the decomposition of the *Wagner* graph; we indicate the number of crossings in the parenthesis. **(a)** Force-based drawing of the graph, **(b)** DP (3), **(c)** `Global conGreedy+` (8), **(d)** `Local conGreedy+` (9), **(e)** `Local conGreedy+ & LS` (5), and **(f)** `Global conGreedy+ & LS` (4).

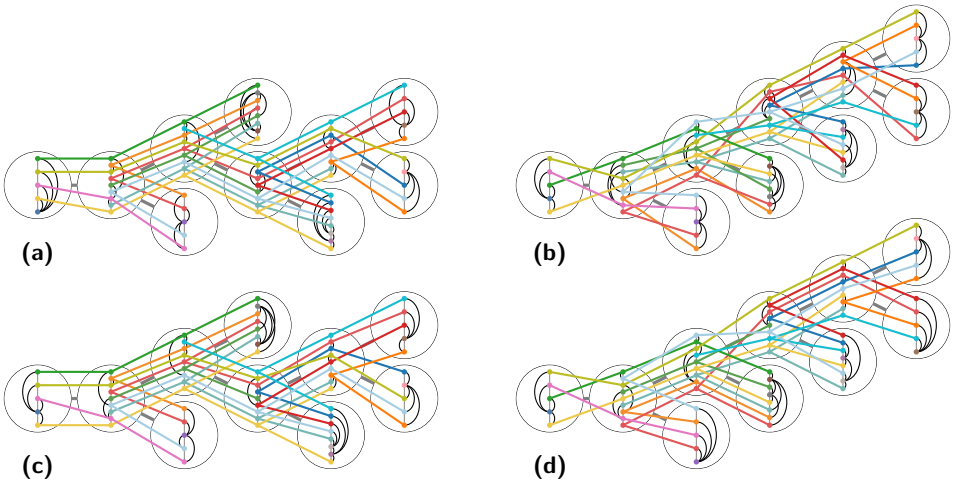


Figure 11: Comparison of the drawings computed by our heuristics for the *Brinkmann* graph[4]; we indicate the number of crossings in the parenthesis. **(a)** `Global conGreedy+` (113), **(b)** `Local conGreedy+` (105), **(c)** `Global conGreedy+ & LS` (64), and **(d)** `Local conGreedy+ & LS` (72).

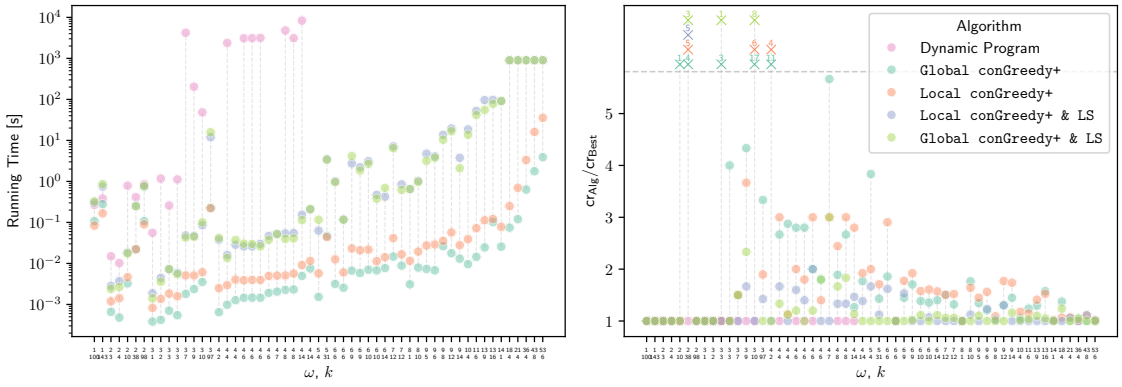


Figure 12: **(a)** Wall-clock running time ($\log_{10}$-scale) of the heuristics and **(b)** their optimality-ratio compared to the best found solution for different values of ω and number of bags k. We group instances based on ω, sort each group by k, and indicate on the x-axis the value of ω and k in the first and second row, respectively. Vertical lines connect data points for one instance.

for instances with larger width, `Global conGreedy+` seems to work slightly better than `Local conGreedy+`. We believe that computing a book drawing for each G_i individually could yield situations where the obtained spine orders yield many `t`/`t`-crossings. In particular, we suspect that spine orders that "locally" admit few crossings could "globally" introduce, for example, unexpected "criss-cross" `t`/`t`-crossings (recall Figure 6d). We see the strength of our DP for decompositions of width two or three. For larger width, the high running time becomes impractical. However, small-width instances are often simple enough that all heuristics can solve them to optimality in a few milliseconds. Interestingly, for instances with very large treewidth, the benefit of local search drastically decreases: Despite performing movement operations for the entire fifteen minutes, the obtained solution is, if at all, only negligibly better then the corresponding heuristic without the local search. However, for instances of moderate width, local search considerably improved the initial solution.

8 Expert Evaluation

We conducted an expert evaluation to assess the usefulness and visual quality of the three drawing styles studied in this paper along with two drawing styles that visualize the graph G_i, for each $V_i \in T$, as incidence matrix and with an embedding consistent to that of a drawing for the entire graph G, respectively; see the blue marked entries in Section 3. The evaluation was designed as an online questionnaire, beginning with a brief introduction into witness drawings in general and a reminder of the definitions of tree decompositions and treewidth.[5] In the beginning, we informed the participants that they will see five different visualizations of a tree decomposition of a graph G and showed them a node-link drawing and the incidence matrix of G; see Figure 19a in Appendix B. We used the *Krackhardt kite graph* from our dataset for the expert evaluation and showed the participants a small overview of the five different visualizations so that they could get

[5] We remarked that participants should have a basic knowledge on these concepts.

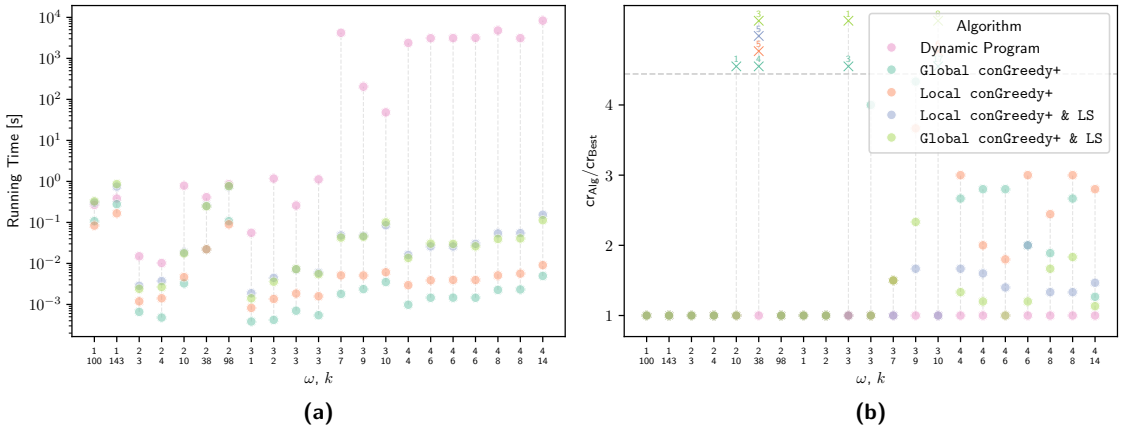


Figure 13: Filtered version of Figure 12, showing only instances for which the DP terminated within six hours.

familiar with them (similar to Figure 19b–f). The linear drawing was created using our dynamic program, and the four remaining ones were created by hand by adapting the linear drawing while ensuring that the number of crossings remains comparable. We showed to the participants one drawing after the other, in a large version and with the node-link drawing of G next to it; see Figure 19b–f in Appendix B for the individual witness drawings. The participants should rate on a 5-point Likert scale how hard or easy they consider performing the following five different tasks with the help of the drawing.

T1) Verifying that each vertex is in at least one bag.

T2) Verifying that for each edge there is a bag containing its endpoints.

T3) Verifying that the bags containing a given vertex form a connected subtree.

T4) Determining the width of the decomposition.

T5) Mapping between the drawings inside the bags and the subdrawings of the graph.

To balance learning effects, we used *Latin Square* to determine the order in which the drawings were shown to the participants. In the end, we asked the participants to rate the five drawings based on visual appeal and indicate the importance they assign to a set of properties such as the number of crossings, or verifiability of the tree decomposition. The full questionnaire is available on OSF [17].

We invited participants of the *International Symposium on Graph Drawing and Network Visualization* 2025 (*GD'25*) to take part in the evaluation. Additionally, we invited Bachelor's, Master's, and PhD students from collaborating research groups with a background in theoretical computer science and/or graph drawing. Overall, twelve persons participated in the evaluation (one Bachelor or Master student, ten PhD students, and one postdoctoral researcher or above). Of these, two considered themselves proficient with tree decompositions and treewidth, eight had previously worked with these concepts and had also seen visual examples, and two participants indicated that they had heard the concepts before but never worked with them. Eight participants indicated

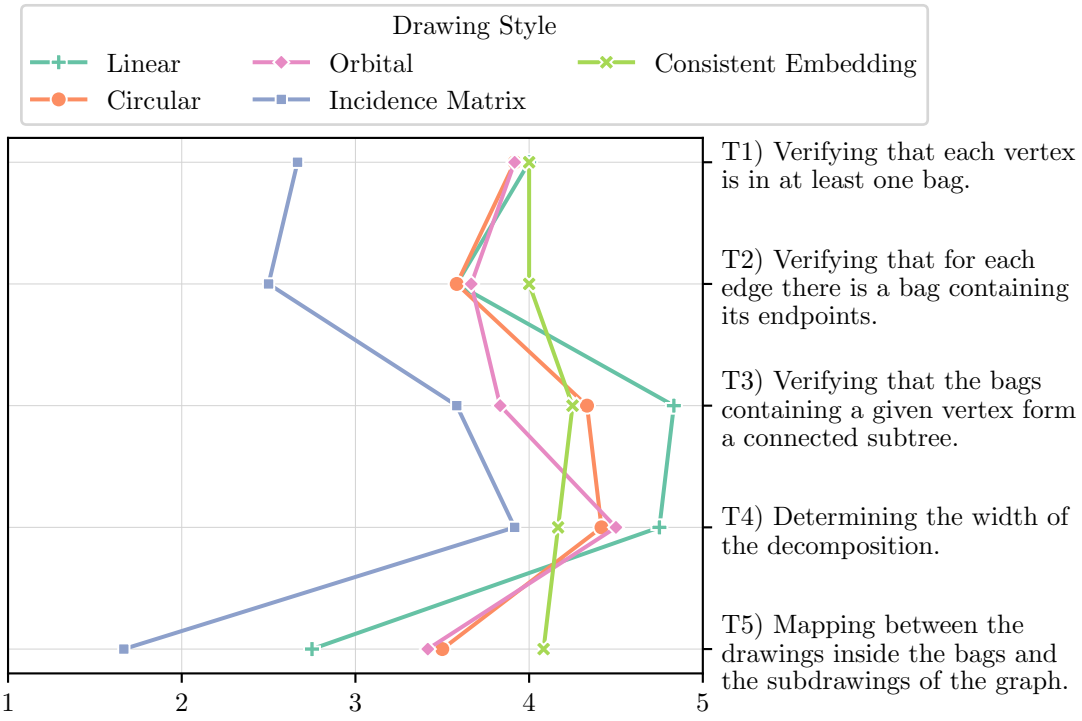


Figure 14: Average scores of the five considered drawing styles. Participants were asked to rate on a 5-point-Likert scale how hard (1) or easy (5) they consider performing each of the tasks T1–T5.

that they work in the area of graph drawing, five in computational geometry and parameterized algorithms, respectively, four in algorithm engineering, and one in information visualization and artificial intelligence each (multiple answers were possible). Below, we analyze the participants' responses in more detail.

Results on Visualization Styles. Recall that we asked participants, for each drawing, how hard or easy they consider performing five different tasks. Figure 14 shows the considered tasks and the corresponding average scores of the responses. In the following, we refer to the tasks with T1–T5; also indicated in Figure 14. In general, we can observe that the visualization using incidence matrices was rated considerably worse than the other styles for all tasks. The remaining styles were rated similarly for T1, T2, and T4. For T1, i.e., verifying that each vertex appears in at least one bag, no significant difference among the four drawing styles was perceivable. For T2, i.e., verifying that each edge appears in at least one bag, the drawing style that uses the consistent embedding was preferred. Interestingly, for T2, this style was also rated notably better than the three drawing styles studied in this paper, although the circular and orbital drawings are, at least compared to linear drawings, visually closer to the style that uses a consistent embedding. For verifying the last criterion, i.e., that bags containing a specific vertex form a connected subtree, the orbital drawing style was rated worse than the other three styles. We suspect that forcing the tracks to stay within the track routing area is the main reason for this outcome. In particular,

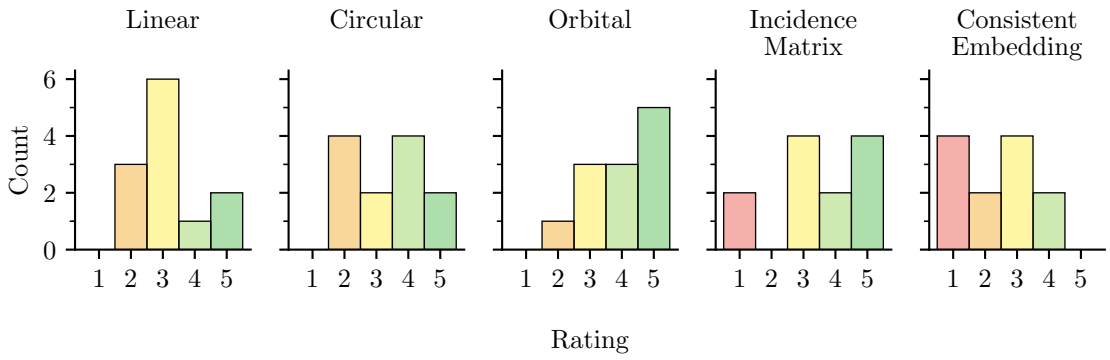


Figure 15: Summary of the ratings for the visual quality (1 is the worst score, 5 the best), indicating how often each style received which rating.

using a track routing area can result in visual clutter that can hinder the participants in following individual tracks; compare also Figure 14d to the other drawing styles in Figure 14. On the other hand, we believe that a vertical alignment of vertices inside the disks such as in linear drawings facilitated the completion of this task, resulting in higher scores for the linear style for T3. For T4, i.e., validating the width of the decomposition, the style that uses incidence matrices was rated the worst. This is surprising since the size of the matrix should be a salient feature to detect and examine the largest bag. With T5, we aimed at evaluating the recognizability of graph features in the drawing of the decomposition. As expected, the style that uses consistent embeddings was rated best for this task and the one that uses the incidence matrix by far the worst. Observe that the matrix-based drawing was rated worst across nearly all tasks, which could also be attributed to a general unfamiliarity with matrix-style visualizations compared to node-link diagrams.

We also asked participants to rate the visual quality of the five drawings from 1 (worst) to 5 (best). Figure 15 shows the results. From top to bottom, the participants ranked the styles as follows: orbital drawing (4.00 on average), style based on incidence matrices (3.50), circular drawing (3.33), linear drawing (3.17), and the worst the style with the consistent embedding (2.33). We can observe that orbital drawings and those using the incident matrix received, on average, the highest scores. Thus, the perceived visual quality does not seem to correlate positively with the perceived difficulty in performing tasks; compare Figures 14 and 15. In particular, recall that the matrix-based style was rated worst across nearly all tasks, and the orbital style often was rated worse than drawings using a linear, circular, or consistent arrangement of the vertices. For orbital drawings, we believe that this outcome is mainly influenced by the track routing area; compare also circular and orbital drawings in Figure 15. The style with the consistent embedding received the lowest scores on average, despite receiving good rates in the task-related questions; recall Figure 14. Half of the participants selected the middle rating for linear drawings.

Results on the Importance of Properties. Finally, we asked participants to judge the importance of the properties listed in Figure 16. This time, we explicitly named the options, ranging from *Not Important* (we associate to this a score of 1) to *Very Important* (score 5). In the following, we refer to the properties as P1–P5; also indicated in Figure 16. Using above-mentioned scores, we obtain an average importance rating of 4.17 for P1, 3.75 for P2, 3.83 for P3, 3.08 for P4, and

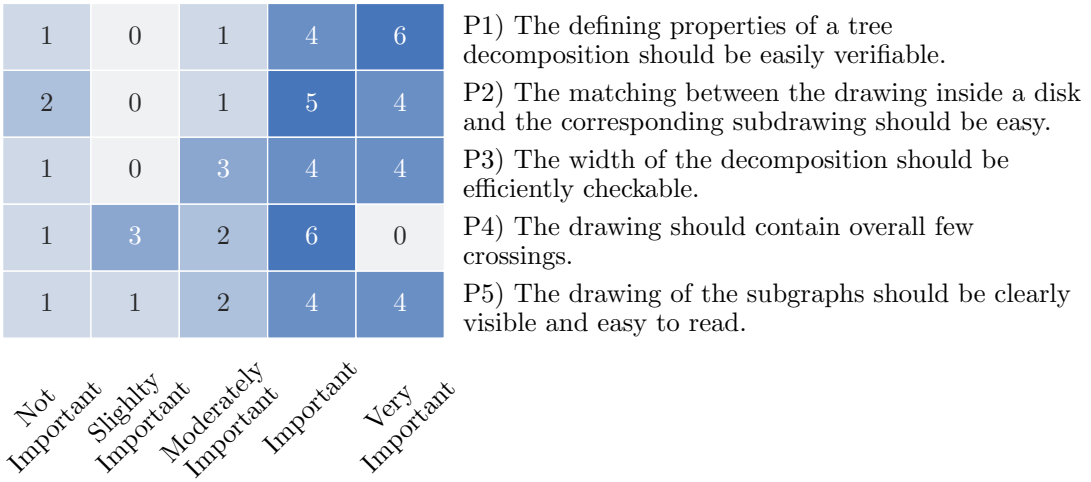


Figure 16: Summary of the indicated importance of the five properties. The numbers in the cells indicate how often each property received which importance score. We additionally color-code the numerical value using different shades of blue.

3.75 for P5. Every property was rated by at least half of the participants as important or very important. Since our drawings should serve as a visual certificate for bounded treewidth, P1 and P3 targeted the verifiability of the defining properties of a bounded-width tree decomposition. Being able to verify that the decomposition is valid (P1) was considered important (or very important) by almost all participants. Only for one participant this was not important at all. However, for this person all properties except P3 were rated as unimportant. Regarding the verifiability of the width of the decomposition, i.e., P3, one participant indicated that this was not important and three indicated that it was only moderately important. Compared to P1, fewer thought that this was very important. Overall, nine people indicated that P2 was (very) important, which is more than P3, P4, and P5, i.e., being able to verify the width of the decomposition, having few crossings in the drawing, and a readable drawing of the subgraphs G_i. This, in particular, means that ensuring consistency across the drawings in the bags was indicated as more important than minimizing the crossings. One participant even commented that verifying whether each edge is part of at least one bag, i.e., T2 in Figure 14, was generally tedious, but easier when the drawings made clear that the bags induced a cover of the entire graph. Together with the responses to P5, this suggests that, when creating witness drawings, one should prioritize the structure of the drawings in the bags over minimizing the number of crossings. In general, the responses for P2 and P5 indicate that information faithfulness is important when designing witness drawings.

Concluding Discussion. Altogether, this expert evaluation partly confirms our design choices for witness drawings and the considered drawing styles. Furthermore, the results indicate that witness drawings can aid users in verifying that a given tree decomposition is valid[6] and that the treewidth of a graph G is bounded. On the other hand, the expert evaluation revealed that crossing

[6]In fact, one participant even correctly noted that one track edge was missing in a preliminary version of Figure 14e, which demonstrates a profound understanding of the semantics of our drawings.

minimization, while being considered important for the readability in graph drawing [64], might not be the primary optimization criterion for the studied witness drawings. Instead, we should aim for a consistent representation of the graphs across bags and with respect to a drawing of the entire graph G (if there is one). In particular, for such combined visualizations, i.e., drawings of the tree decomposition and G, we might face conflicting optimization criteria: Consistent drawings in the decomposition could result in a low-quality drawing of G. Especially for decompositions of larger width, this optimization criterion needs further investigations. For large-width decompositions, we can no longer rely on color as an additional visual variable to differentiate vertices in the bags, as also noted by one participant. Hence, we may want to investigate additional visual encodings for such drawings. Finally, participating experts also commented that the current visualizations are targeted on verifying that each edge is in at least one bag and every vertex induces a connected subtree, but less on the condition that every vertex is in at least one bag or the width of the decomposition. This underlines the potential for exploring other, additional, design elements to convey these properties and/or interactive witness drawings.

9 Concluding Remarks and Future Directions

In this paper, we have initiated the study of visualizing tree and path decompositions of graphs to visually certify their treewidth and pathwidth, respectively. We have explored the design space of such witness drawings and further examined three styles from an algorithmic perspective. Our experimental study shows that for decompositions of width at most three, we can compute exact crossing-minimal drawings in a few seconds. For decompositions of medium to larger width, the heuristics combined with a local search step seem to be the preferred method. We see the computation of a crossing-minimal drawing where we do not restrict the placement of the child bags or the starting angle inside the disk as a challenging open algorithmic problem. Note that our exact algorithms can be adapted to handle a discrete set of candidate starting angles α but the problem of optimizing this parameter remains open.

Our expert evaluation confirms that the three studied drawing styles can aid in verifying that a given tree decomposition is valid. At the same time, the evaluation also shows that crossing minimization is not necessarily the primary optimization criterion to aim for. Consequently, we see the exploration of well-defined optimization criteria that result in a consistent appearance of the drawings across bags as an important direction for future work. Moreover, our description of the design space in Section 3 should be seen as an invitation to develop further witness-drawing styles along with corresponding exact and heuristic algorithms to compute them. The empirical and experimental evaluation of these new width-witness drawings is also an interesting open problem. In particular, we see a larger user study measuring actual task performance as a natural next step. Finally, designing and computing suitable witness drawings for other width parameters of graphs is an interesting, independent direction for future work.

Acknowledgments

We thank all participants for taking part in the expert evaluation. Furthermore, we thank the anonymous reviewers for their valuable comments.

References

[1] L. Angori, W. Didimo, F. Montecchiani, D. Pagliuca, and A. Tappini. Chordlink: A new hybrid visualization model. In D. Archambault and C. D. Tóth, editors, *Proc. 27th International Symposium on Graph Drawing and Network Visualization (GD'19)*, volume 11904 of *Lecture Notes in Computer Science*, pages 276–290. Springer, 2019. doi:10.1007/978-3-030-35802-0_22.

[2] E. N. Argyriou, M. A. Bekos, M. Kaufmann, and A. Symvonis. On metro-line crossing minimization. *Journal of Graph Algorithms and Applications*, 14(1):75–96, 2010. doi:10.7155/jgaa.00199.

[3] B. Aronov, M. Dulieu, and F. Hurtado. Witness (Delaunay) graphs. *Computational Geometry*, 44(6):329–344, 2011. doi:10.1016/j.comgeo.2011.01.001.

[4] B. Aronov, M. Dulieu, and F. Hurtado. Witness rectangle graphs. In F. Dehne, J. Iacono, and J. Sack, editors, *Proc 12th International Symposium on Algorithms and Data Structures (WADS'11)*, volume 6844 of *Lecture Notes in Computer Science*, pages 73–85. Springer, 2011. doi:10.1007/978-3-642-22300-6_7.

[5] B. Aronov, M. Dulieu, and F. Hurtado. Witness gabriel graphs. *Computational Geometry*, 46(7):894–908, 2013. doi:10.1016/j.comgeo.2011.06.004.

[6] B. Aronov, M. Dulieu, and F. Hurtado. Witness rectangle graphs. *Graphs and Combinatorics*, 30(4):827–846, 2014. doi:10.1007/s00373-013-1316-x.

[7] M. Asquith, J. Gudmundsson, and D. Merrick. An ILP for the metro-line crossing problem. In J. Harland and P. Manyem, editors, *Proc. 14th Computing: The Australasian Theory Symposium (CATS'08)*, volume 77 of *CRPIT*, pages 49–56. Australian Computer Society, 2008.

[8] M. J. Bannister and D. Eppstein. Crossing minimization for 1-page and 2-page drawings of graphs with bounded treewidth. In C. Duncan and A. Symvonis, editors, *Proc. 22nd International Symposium on Graph Drawing and Network Visualization (GD'14)*, volume 8871 of *Lecture Notes in Computer Science*, pages 210–221. Springer, 2014. doi:10.1007/978-3-662-45803-7_18.

[9] L. Barth, B. Niedermann, I. Rutter, and M. Wolf. A topology-shape-metrics framework for ortho-radial graph drawing. *Discrete & Computational Geometry*, 70(4):1292–1355, 2023. doi:10.1007/s00454-023-00593-y.

[10] M. Baur and U. Brandes. Crossing reduction in circular layouts. In J. Hromkovič, M. Nagl, and B. Westfechtel, editors, *Proc. 30th International Workshop on Graph-Theoretic Concepts in Computer Science (WG'04)*, volume 3353 of *LNCS*, pages 332–343. Springer Berlin Heidelberg, 2004. doi:10.1007/978-3-540-30559-0_28.

[11] M. A. Bekos, M. Kaufmann, K. Potika, and A. Symvonis. Line crossing minimization on metro maps. In S. Hong, T. Nishizeki, and W. Quan, editors, *Proc. 15th International Symposium on Graph Drawing and Network Visualization (GD'07)*, volume 4875 of *Lecture Notes in Computer Science*, pages 231–242. Springer, 2007. doi:10.1007/978-3-540-77537-9_24.

[12] M. Benkert, M. Nöllenburg, T. Uno, and A. Wolff. Minimizing intra-edge crossings in wiring diagrams and public transportation maps. In M. Kaufmann and D. Wagner, editors, *Proc. 14th International Symposium on Graph Drawing and Network Visualization (GD'06)*, volume 4372 of *Lecture Notes in Computer Science*, pages 270–281. Springer, 2006. doi:10.1007/978-3-540-70904-6_27.

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

[14] H. L. Bodlaender. Treewidth: Characterizations, applications, and computations. In F. V. Fomin, editor, *Proc. 32nd International Workshop on Graph-Theoretic Concepts in Computer Science (WG'06)*, volume 4271 of *Lecture Notes in Computer Science*, pages 1–14. Springer, 2006. doi:10.1007/11917496_1.

[15] H. L. Bodlaender and T. Kloks. Efficient and constructive algorithms for the pathwidth and treewidth of graphs. *Journal of Algorithms*, 21(2):358–402, 1996. doi:10.1006/jagm.1996.0049.

[16] A. Bonerath, M. Nöllenburg, S. Terziadis, M. Wallinger, and J. Wulms. Boundary labeling in a circular orbit. In S. Felsner and K. Klein, editors, *Proc. 32nd International Symposium on Graph Drawing and Network Visualization (GD'24)*, volume 320 of *LIPIcs*, pages 22:1–22:17. Schloss Dagstuhl – Leibniz-Zentrum für Informatik, 2024. doi:10.4230/LIPICS.GD.2024.22.

[17] A. Chiu, T. Depian, D. Eppstein, M. T. Goodrich, and M. Nöllenburg. Visualizing treewidth: Supplementary material, 2025. Source Code. doi:10.17605/OSF.IO/QFZ5V.

[18] M. Cygan, F. V. Fomin, L. Kowalik, D. Lokshtanov, D. Marx, M. Pilipczuk, M. Pilipczuk, and S. Saurabh. *Parameterized Algorithms*. Springer, 2015. doi:10.1007/978-3-319-21275-3.

[19] G. Di Battista, P. Eades, R. Tamassia, and I. G. Tollis. *Graph Drawing: Algorithms for the Visualization of Graphs*. Prentice Hall, 1999.

[20] G. Di Battista and R. Tamassia. On-line maintenance of triconnected components with SPQR-trees. *Algorithmica*, 15(4):302–318, 1996. doi:10.1007/s004539900017.

[21] M. Dickerson, D. Eppstein, M. T. Goodrich, and J. Y. Meng. Confluent drawings: Visualizing non-planar diagrams in a planar way. In G. Liotta, editor, *Proc. 11th International Symposium on Graph Drawing and Network Visualization (GD'03)*, volume 2912 of *Lecture Notes in Computer Science*, pages 1–12. Springer, 2004. doi:10.1007/978-3-540-24595-7_1.

[22] M. Dickerson, D. Eppstein, M. T. Goodrich, and J. Y. Meng. Confluent drawings: Visualizing non-planar diagrams in a planar way. *Journal of Graph Algorithms and Applications*, 9(1):31–52, 2005. doi:10.7155/jgaa.00099.

[23] W. Didimo, G. Liotta, and F. Montecchiani. A survey on graph drawing beyond planarity. *ACM Computing Surveys*, 52(1):4:1–4:37, 2019. doi:10.1145/3301281.

[24] A. Dobler, M. Jünger, P. J. Jünger, J. Meffert, P. Mutzel, and M. Nöllenburg. Revisiting ILP models for exact crossing minimization in storyline drawings. In S. Felsner and K. Klein, editors, *Proc. 32nd International Symposium on Graph Drawing and Network Visualization (GD'24)*, volume 320 of *LIPIcs*, pages 31:1–31:19. Schloss Dagstuhl – Leibniz-Zentrum für Informatik, 2024. doi:10.4230/LIPIcs.GD.2024.31.

[25] V. Dujmović, D. Eppstein, and D. R. Wood. Structure of graphs with locally restricted crossings. *SIAM Journal on Discrete Mathematics*, 31(2):805–824, 2017. doi:10.1137/16M1062879.

[26] V. Dujmović, P. Morin, and D. R. Wood. Layout of graphs with bounded tree-width. *SIAM Journal on Computing*, 34(3):553–579, 2005. doi:10.1137/S0097539702416141.

[27] V. Dujmović and D. R. Wood. On linear layouts of graphs. *Discrete Mathematics & Theoretical Computer Science*, 6(2):339–358, 2004. doi:10.46298/DMTCS.317.

[28] J. Ellson, E. R. Gansner, E. Koutsofios, S. C. North, and G. Woodhull. Graphviz - open source graph drawing tools. In P. Mutzel, M. Jünger, and S. Leipert, editors, *Proc. 9th International Symposium on Graph Drawing and Network Visualization (GD'01)*, volume 2265 of *Lecture Notes in Computer Science*, pages 483–484. Springer, 2001. doi:10.1007/3-540-45848-4_57.

[29] D. Eppstein. Diameter and treewidth in minor-closed graph families. *Algorithmica*, 27(3):275–291, 2000. doi:10.1007/s004530010020.

[30] D. Eppstein. Separating thickness from geometric thickness. In M. T. Goodrich and S. G. Kobourov, editors, *Proc. 10th International Symposium on Graph Drawing and Network Visualization (GD'02)*, volume 2528 of *Lecture Notes in Computer Science*, pages 150–162. Springer, 2002. doi:10.1007/3-540-36151-0_15.

[31] D. Eppstein. Tree decomposition.svg via Wikimedia Commons, 2010. Last accessed: 2025-08-14. URL: https://upload.wikimedia.org/wikipedia/commons/a/a7/Tree_decomposition.svg.

[32] D. Eppstein. What is ... treewidth? *Notices of the AMS*, 72(2):172–175, 2025. doi:10.1090/noti3043.

[33] D. Eppstein, D. Holten, M. Löffler, M. Nöllenburg, B. Speckmann, and K. Verbeek. Strict confluent drawing. In S. K. Wismath and A. Wolff, editors, *Proc. 21st International Symposium on Graph Drawing and Network Visualization (GD'13)*, volume 8242 of *Lecture Notes in Computer Science*, pages 352–363. Springer, 2013. doi:10.1007/978-3-319-03841-4_31.

[34] D. Eppstein, D. Holten, M. Löffler, M. Nöllenburg, B. Speckmann, and K. Verbeek. Strict confluent drawing. *Journal of Computational Geometry*, 7(1):22–46, 2016. doi:10.20382/JOCG.V7I1A2.

[35] M. Fink and S. Pupyrev. Metro-line crossing minimization: Hardness, approximations, and tractable cases. In S. Wismath and A. Wolff, editors, *Proc. 21st International Symposium on Graph Drawing and Network Visualization (GD'13)*, volume 8242 of *Lecture Notes in Computer Science*, pages 328–339. Springer, 2013. doi:10.1007/978-3-319-03841-4_29.

[36] H. Förster, F. Klesen, T. Dwyer, P. Eades, S. Hong, S. G. Kobourov, G. Liotta, K. Misue, F. Montecchiani, A. Pastukhov, and F. Schreiber. GraphTrials: Visual proofs of graph properties. In S. Felsner and K. Klein, editors, *Proc. 32nd International Symposium on Graph Drawing and Network Visualization (GD'24)*, volume 320 of *LIPIcs*, pages 16:1–16:18. Schloss Dagstuhl - Leibniz-Zentrum für Informatik, 2024. doi:10.4230/LIPIcs.GD.2024.16.

[37] T. M. J. Fruchterman and E. M. Reingold. Graph drawing by force-directed placement. *Software: Practice and Experience*, 21(11):1129–1164, 1991. doi:10.1002/spe.4380211102.

[38] H. Förster, R. Ganian, F. Klute, and M. Nöllenburg. On strict (outer-)confluent graphs. In D. Archambault and C. D. Tóth, editors, *Proc. 27th International Symposium on Graph Drawing and Network Visualization (GD'19)*, volume 11904 of *Lecture Notes in Computer Science*, pages 147–161. Springer, 2019. doi:10.1007/978-3-030-35802-0_12.

[39] H. Förster, R. Ganian, F. Klute, and M. Nöllenburg. On strict (outer-)confluent graphs. *Journal of Graph Algorithms and Applications*, 25(1):481–512, 2021. doi:10.7155/jgaa.00568.

[40] H. Förster, F. Klesen, T. Dwyer, P. Eades, S.-H. Hong, S. Kobourov, G. Liotta, K. Misue, F. Montecchiani, A. Pastukhov, and F. Schreiber. Graphtrials: Visual proofs of graph properties. *IEEE Transactions on Visualization and Computer Graphics*, 31(10):8767–8781, 2025. doi:10.1109/tvcg.2025.3577533.

[41] E. R. Gansner and Y. Koren. Improved circular layouts. In M. Kaufmann and D. Wagner, editors, *Proc. 14th International Symposium on Graph Drawing and Network Visualization (GD'06)*, volume 4372 of *Lecture Notes in Computer Science*, pages 386–398. Springer, 2006. doi:10.1007/978-3-540-70904-6_37.

[42] M. Gronemann, M. Jünger, F. Liers, and F. Mambelli. Crossing minimization in storyline visualization. In Y. Hu and M. Nöllenburg, editors, *Proc. 24th International Symposium on Graph Drawing and Network Visualization (GD'16)*, volume 9801 of *Lecture Notes in Computer Science*, pages 367–381. Springer, 2016. doi:10.1007/978-3-319-50106-2_29.

[43] N. Henry, J.-D. Fekete, and M. J. McGuffin. Nodetrix: a hybrid visualization of social networks. *IEEE Transactions on Visualization and Computer Graphics*, 13(6):1302–1309, 2007. doi:10.1109/tvcg.2007.70582.

[44] M. Hoffmann, L. Kettner, and S. Näher. Two computational geometry libraries: LEDA and CGAL. In *Handbook of Discrete and Computational Geometry*, pages 1799–1831. Chapman and Hall/CRC, 2017.

[45] M. Jünger and P. Mutzel. *Graph Drawing Software*. Springer, 2012.

[46] M. Kaufmann and D. Wagner, editors. *Drawing Graphs: Methods and Models*, volume 2025 of *Lecture Notes in Computer Science*. Springer, 2001. doi:10.1007/3-540-44969-8.

[47] J. Klawitter, T. Mchedlidze, and M. Nöllenburg. Experimental evaluation of book drawing algorithms. In F. Frati and K. Ma, editors, *Proc. 25th International Symposium on Graph Drawing and Network Visualization (GD'17)*, volume 10692 of *Lecture Notes in Computer Science*, pages 224–238. Springer, 2017. doi:10.1007/978-3-319-73915-1_19.

[48] E. Korach and N. Solel. Tree-width, path-width, and cutwidth. *Discrete Applied Mathematics*, 43(1):97–101, 1993. doi:10.1016/0166-218X(93)90171-J.

[49] I. Kostitsyna, M. Nöllenburg, V. Polishchuk, A. Schulz, and D. Strash. On minimizing crossings in storyline visualizations. In E. D. Giacomo and A. Lubiw, editors, *Proc. 23rd International Symposium on Graph Drawing and Network Visualization (GD'15)*, volume 9411 of *Lecture Notes in Computer Science*, pages 192–198. Springer, 2015. doi:10.1007/978-3-319-27261-0_16.

[50] D. Kratsch, R. M. McConnell, K. Mehlhorn, and J. P. Spinrad. Certifying algorithms for recognizing interval graphs and permutation graphs. In *Proc. 14th ACM-SIAM Symposium on Discrete Algorithms (SODA'03)*, pages 158–167. ACM/SIAM, 2003.

[51] D. Kratsch, R. M. McConnell, K. Mehlhorn, and J. P. Spinrad. Certifying algorithms for recognizing interval graphs and permutation graphs. *SIAM Journal on Computing*, 36(2):326–353, 2006. doi:10.1137/S0097539703437855.

[52] W. J. Lenhart and G. Liotta. Mutual witness Gabriel drawings of complete bipartite graphs. *Theoretical Computer Science*, 974:114115, 2023. doi:10.1016/j.tcs.2023.114115.

[53] W. J. Lenhart and G. Liotta. Mutual witness gabriel drawings of complete bipartite graphs. In P. Angelini and R. von Hanxleden, editors, *Proc. 30th Proc. 21st International Symposium on Graph Drawing and Network Visualization (GD'22)*, volume 13764 of *Lecture Notes in Computer Science*, pages 25–39. Springer, 2023. doi:10.1007/978-3-031-22203-0_3.

[54] S. Maniu, P. Senellart, and S. Jog. An experimental study of the treewidth of real-world graph data. In P. Barceló and M. Calautti, editors, *Proc. 22nd International Conference on Database Theory (ICDT'19)*, volume 127 of *LIPIcs*, pages 12:1–12:18. Schloss Dagstuhl – Leibniz-Zentrum für Informatik, 2019. doi:10.4230/LIPICS.ICDT.2019.12.

[55] S. Masuda, T. Kashiwabara, K. Nakajima, and T. Fujisawa. On the NP-completeness of a computer network layout problem. In *Proc. IEEE Intl. Symposium on Circuits and Systems*, pages 292–295, 1987.

[56] J. Matoušek and R. Thomas. On the complexity of finding iso- and other morphisms for partial k-trees. *Discrete Mathematics*, 108(1–3):343–364, 1992. doi:10.1016/0012-365x(92)90687-b.

[57] R. M. McConnell, K. Mehlhorn, S. Näher, and P. Schweitzer. Certifying algorithms. *Computer Science Review*, 5(2):119–161, 2011. doi:10.1016/j.cosrev.2010.09.009.

[58] K. Mehlhorn, S. Näher, T. Schilz, S. Schirra, M. Seel, R. Seidel, and C. Uhrig. Checking geometric programs or verification of geometric structures. In S. Whitesides, editor, *Proc. 12th International Symposium on Computational Geometry (SoCG'96)*, SCG '96, pages 159–165. ACM, 1996. doi:10.1145/237218.237344.

[59] K. Mehlhorn, S. Näher, M. Seel, R. Seidel, T. Schilz, S. Schirra, and C. Uhrig. Checking geometric programs or verification of geometric structures. *Computational Geometry*, 12(1-2):85–103, 1999. doi:10.1016/S0925-7721(98)00036-4.

[60] P. Mutzel. The SPQR-tree data structure in graph drawing. In J. C. M. Baeten, J. K. Lenstra, J. Parrow, and G. J. Woeginger, editors, *Proc. 30th International Colloquium on Automata, Languages and Programming (ICALP'03)*, volume 2719 of *Lecture Notes in Computer Science*, pages 34–46. Springer, 2003. doi:10.1007/3-540-45061-0_4.

[61] M. Nöllenburg. An improved algorithm for the metro-line crossing minimization problem. In D. Eppstein and E. R. Gansner, editors, *Proc. 17th International Symposium on Graph Drawing and Network Visualization (GD'09)*, volume 5849 of *Lecture Notes in Computer Science*, pages 381–392. Springer, 2009. doi:10.1007/978-3-642-11805-0_36.

[62] S. Pupyrev, L. Nachmanson, S. Bereg, and A. E. Holroyd. Edge routing with ordered bundles. In M. J. van Kreveld and B. Speckmann, editors, *Proc. 19th International Symposium on Graph Drawing and Network Visualization (GD'11)*, volume 7034 of *Lecture Notes in Computer Science*, pages 136–147. Springer, 2012. doi:10.1007/978-3-642-25878-7_14.

[63] S. Pupyrev, L. Nachmanson, S. Bereg, and A. E. Holroyd. Edge routing with ordered bundles. *Computational Geometry*, 52:18–33, 2016. doi:10.1016/j.comgeo.2015.10.005.

[64] H. C. Purchase. Which aesthetic has the greatest effect on human understanding? In G. D. Battista, editor, *Proc. 5th International Symposium on Graph Drawing and Network Visualization (GD'97)*, volume 1353 of *Lecture Notes in Computer Science*, pages 248–261. Springer, 1997. doi:10.1007/3-540-63938-1_67.

[65] H. C. Purchase, B. Plimmer, R. Baker, and C. Pilcher. Graph drawing aesthetics in user-sketched graph layouts. In C. Lutteroth and P. R. Calder, editors, *Proc. 11th Australasian User Interface Conference (AUIC'10)*, volume 106 of *CRPIT*, pages 80–88. Australian Computer Society, 2010.

[66] M. Schaefer. The graph crossing number and its variants: A survey. *The Electronic Journal of Combinatorics*, 1000, 2013. Version 8. doi:10.37236/2713.

[67] H. Schulz. Treevis.net: A tree visualization reference. *IEEE Computer Graphics and Applications*, 31(6):11–15, 2011. doi:10.1109/MCG.2011.103.

[68] R. Tamassia, editor. *Handbook of Graph Drawing and Visualization*. Chapman and Hall/CRC, 2013.

[69] T. C. van Dijk, M. Fink, N. Fischer, F. Lipp, P. Markfelder, A. Ravsky, S. Suri, and A. Wolff. Block crossings in storyline visualizations. *Journal of Graph Algorithms and Applications*, 21(5):873–913, 2017. doi:10.7155/jgaa.00443.

[70] T. C. van Dijk, F. Lipp, P. Markfelder, and A. Wolff. Computing storyline visualizations with few block crossings. In F. Frati and K. Ma, editors, *Proc. 25th International Symposium on Graph Drawing and Network Visualization (GD'17)*, volume 10692 of *Lecture Notes in Computer Science*, pages 365–378. Springer, 2017. doi:10.1007/978-3-319-73915-1_29.

[71] C. Ware. *Chapter 6 – Static and Moving Patterns*, chapter 6, page 179–237. Elsevier, 2013. doi:10.1016/b978-0-12-381464-7.00006-5.

[72] C. Ware. *Information Visualization: Perception for Design*. Elsevier, 3rd edition, 2013. doi:10.1016/c2009-0-62432-6.

[73] M. Wattenberg. Arc diagrams: Visualizing structure in strings. In P. C. Wong and K. Andrews, editors, *Proc. 8th IEEE Information Visualization Conference (InfoVis'02)*, pages 110–116. IEEE Computer Society, 2002. doi:10.1109/INFVIS.2002.1173155.

A Sample Drawings

We provide in Figures 17 and 18 further witness drawings computed by our algorithms. Figure 17 shows four drawings computed by the DP. Figure 18 shows the drawings for four decompositions computed by `Global conGreedy+ & LS` and `Local conGreedy+ & LS`. We provide for all drawings the number of crossings in the parenthesis. For further information on the respective graphs, we refer to the overview page of *Sage*[7], on which also the dataset is based on.

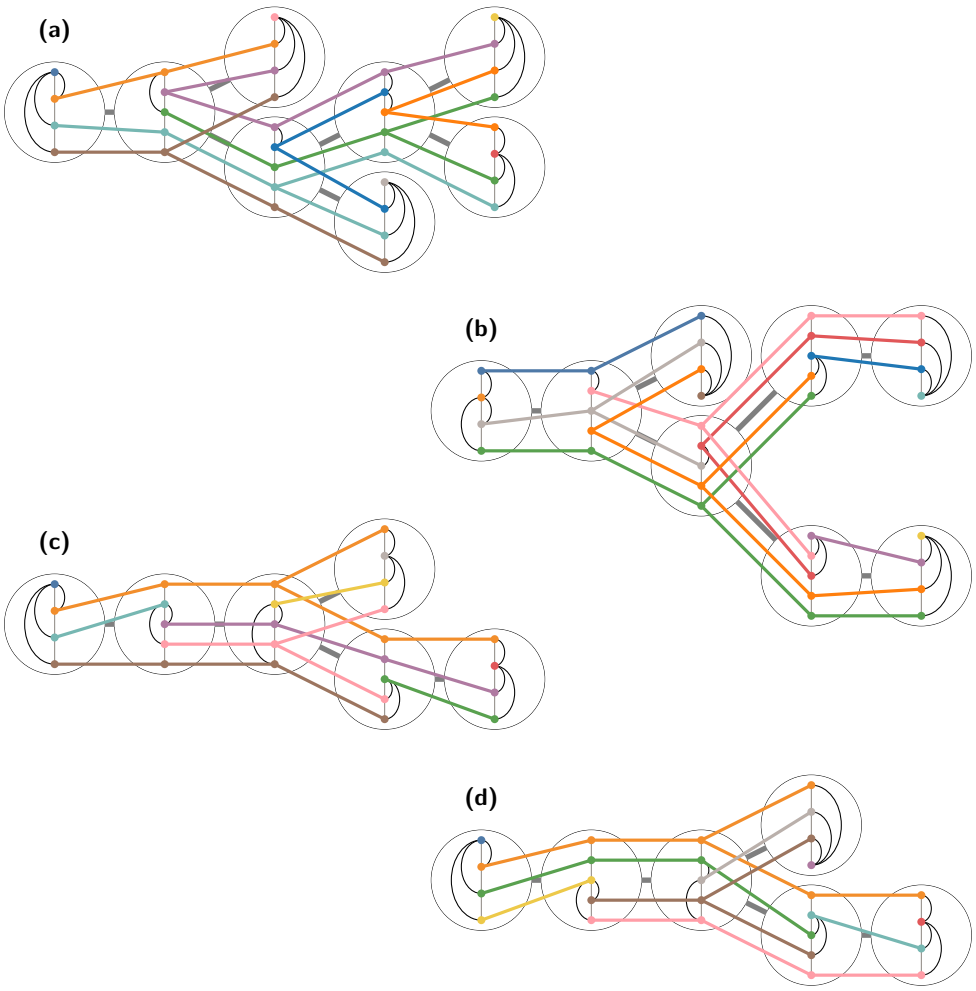


Figure 17: Sample drawings computed by our DP for decompositions of various graphs; we indicate the number of crossings in the parenthesis. **(a)** Bidiakis Cube (6), **(b)** Franklin Graph (9), **(c)** Odd Graph 3 (5), and **(d)** Petersen Graph (5).

[7] https://doc.sagemath.org/html/en/reference/graphs/sage/graphs/graph_generators.html

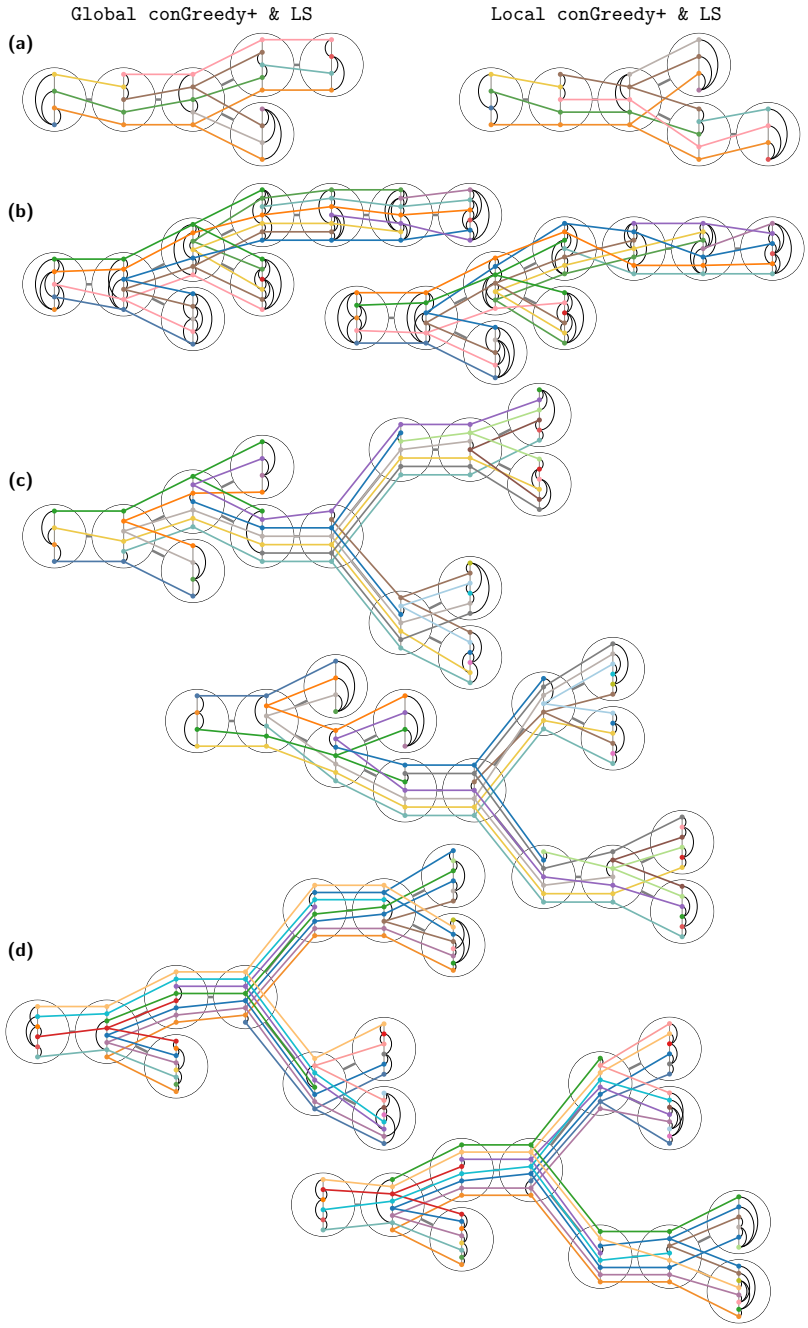


Figure 18: Sample drawings computed by the heuristics combined with the local search for decompositions of various graphs; we indicate the number of crossings in the parenthesis. A vertical bar (`Global conGreedy+ & LS`|`Local conGreedy+ & LS`) separates the number of crossings for the two algorithms. **(a)** Petersen Graph (5|7), **(b)** Poussin Graph (52|51), **(c)** Nauru Graph (39|35), and **(d)** Coxeter Graph (53|50).

B Drawings From the Expert Evaluation

Figure 19 shows the drawings that we provided during the expert evaluation. Figure 19a visualizes the graph both as node-link diagram and incidence matrix. Figure 19b–f contain the five witness drawings that we evaluated.

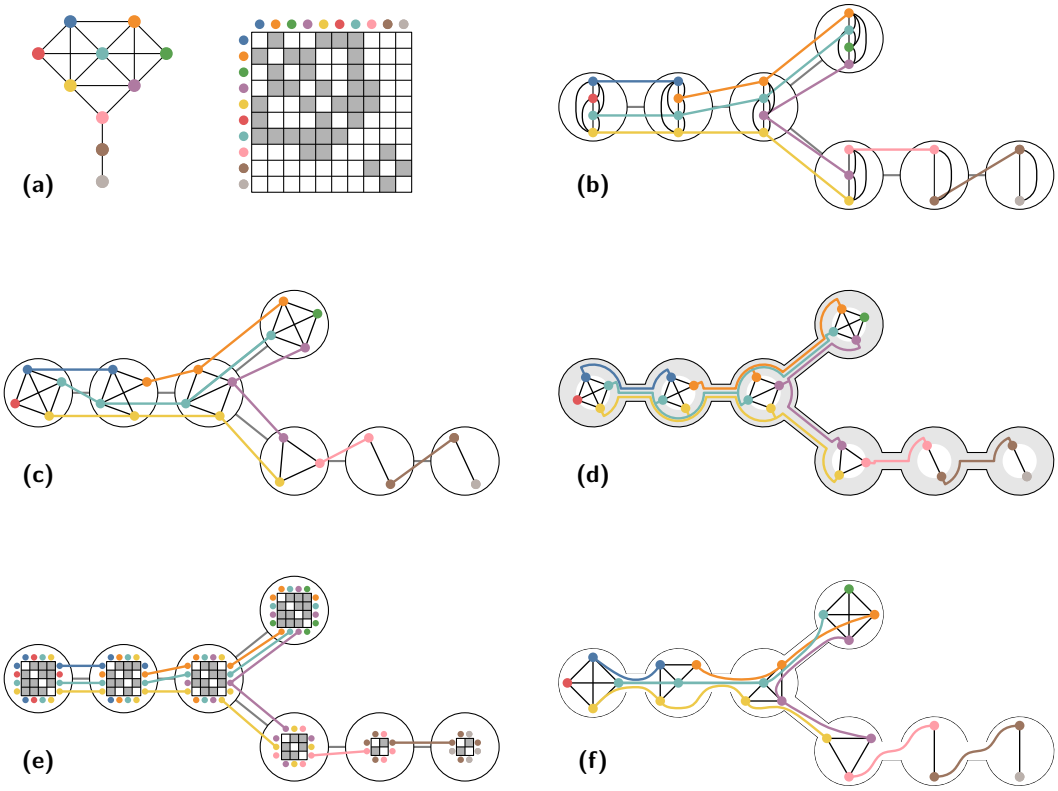


Figure 19: Drawings used in the expert evaluation. **(a)** Graph visualized as node-link diagram and incidence matrix, **(b)** linear drawing, **(c)** circular drawing, **(d)** orbital drawing, **(e)** incidence matrix, and **(f)** consistent embedding.