Using the Metro-Map Metaphor for Drawing Hypergraphs

Fabian Frank[1], Michael Kaufmann[2], Stephen Kobourov[3], Tamara Mchedlidze[4], Sergey Pupyrev[5], Torsten Ueckerdt[1], and Alexander Wolff[6](✉)

[1] Karlsruhe Institute of Technology, Karlsruhe, Germany
fabian-frank97@web.de, torsten.ueckerdt@kit.edu
[2] Universität Tübingen, Tübingen, Germany
mk@uni-tuebingen.de
[3] University of Arizona, Tucson, USA
kobourov@cs.arizona.edu
[4] Utrecht University, Utrecht, Netherlands
mched@iti.uka.de
[5] Facebook, Menlo Park, USA
spupyrev@gmail.com
[6] Universität Würzburg, Würzburg, Germany
alexander.wolff@uni-wuerzburg.de

Abstract. For a planar graph G and a set Π of simple paths in G, we define a *metro-map embedding* to be a planar embedding of G and an ordering of the paths of Π along each edge of G. This definition of a metro-map embedding is motivated by visual representations of hypergraphs using the metro-map metaphor. In a metro-map embedding, two paths cross in a so-called *vertex crossing* if they pass through the vertex and alternate in the circular ordering around it.

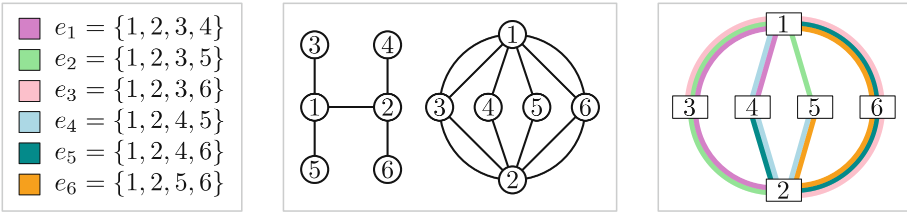
We study the problem of constructing metro-map embeddings with the minimum number of *crossing vertices*, that is, vertices where paths cross. We show that the corresponding decision problem is NP-complete for general planar graphs but can be solved efficiently for trees or if the number of crossing vertices is constant. All our results hold both in a fixed and variable embedding settings.

Keywords: Metro-map metaphor · Hypergraph visualization · Crossing minimization · NP-hardness · Clustered planarity

1 Introduction

We consider a visualization style for hypergraphs that is inspired by schematic metro maps. Such maps are common for urban citizens, who all know that the stations traversed by the same colored curve belong to the same metro line. This intuitive understanding of grouping has been employed to visualize other abstract data forming hypergraphs. For example, Foo [8] turns personal memories into a metro map, Nesbitt [12] and Stott et al. [17] use the metro-map

T. Bureš et al. (Eds.): SOFSEM 2021, LNCS 12607, pp. 361–372, 2021.
https://doi.org/10.1007/978-3-030-67731-2_26



(a) (b) (c)

Fig. 1. (a) The hypergraph H with vertex set $\{1, \ldots, 6\}$ whose hyperedges are all 4-element subsets containing $\{1, 2\}$, (b) two supports of H, and (c) a metro-map drawing of H.

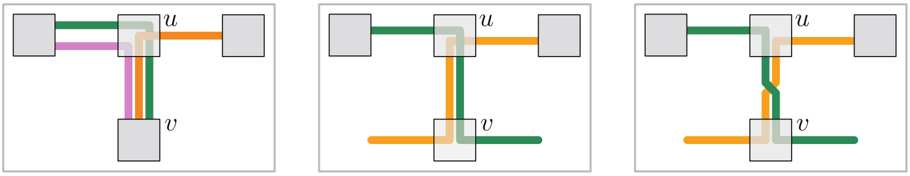
metaphor to visualize relationships between PhD theses and items of a business plan, Sandvad et al. [15] for building Web-based guided tour systems, and Shahaf et al. [16] use it for visualizing historical events. A popular visualization shows the 250 best movies according to a vote on IMDb.com [10].

Informally, the problem of constructing such a visualization for a given hypergraph is as follows. Let $H = (V, \mathcal{E})$ be a hypergraph with vertex set V and edge set $\mathcal{E} \subseteq 2^V$. A *metro-map drawing* of H is a graphical representation where each node in V is depicted by a point in the plane and each hyperedge $e \in \mathcal{E}$ by an open continuous curve, referred to as *line*, that passes through the points corresponding to the vertices in e; see Fig. 4 below.

The problem of constructing a metro-map drawing of a hypergraph can be broken down into several algorithmically challenging steps that are each worth of independent investigation. The first step is to construct a so-called path-based hypergraph support. Given a hypergraph H, a graph G with $V(G) = V(H)$ is a *support* of H if, for each hyperedge e of H, the graph $G[e]$ induced by e is connected. A support is *path-based* if, for every hyperedge e, $G[e]$ contains a Hamiltonian path. For example, Fig. 1 shows a hypergraph H with two supports; the right one is path-based while the left one is not. The second step is to draw the path-based support. The third and final step is to route the lines along the edges of the support. All three steps strongly influence the readability of the final metro-map drawing.

Even just the first step leads to two difficult problems. Constructing a path-based support with the minimum number of edges—a natural optimization goal for obtaining a readable final result—is NP-hard [5]. Assuming that a support G is provided as part of the input, ordering the vertices within a hyperedge e is NP-hard as this task implicitly tests whether the induced subgraph $G[e]$ contains a Hamiltonian path. In this paper, we study the algorithmic complexity of the other two steps: embedding the support and routing the lines.

The input for our problem is a planar graph G (the support of H) and a set Π of simple paths in G (the hyperedges). For example consider the metro-map drawing in the right of Fig. 1. Here the path-based support is $G \cong K_{2,4}$, and the set Π contains for example the path $p_1 = [2, 3, 1, 4]$ for hyperedge



(a) (b) (c)

Fig. 2. The vertex crossing in (a) can be eliminated by reordering lines at v. Sometimes (but not always) a vertex crossing (b) can be turned into a line crossings (c).

$e_1 = \{1, 2, 3, 4\}$. In general, let u be a vertex of G, let $u_1, \ldots, u_k$ be the neighbors of u in clockwise order, as determined by an embedding $\mathcal{G}$ of G, and let $\Pi_i \subseteq \Pi$ be the set of paths that contain the edge (u, u_i). A *line ordering at vertex* u *on the edge* (u, u_i), $\text{ord}_u(u_i)$, is an ordering $p_1, \ldots, p_h$ of Π_i such that u_{i-1} precedes p_1 and p_h precedes u_{i+1} in $\mathcal{G}$. In Fig. 1 the line ordering $\text{ord}_1(3)$ on edge $(1, 3)$ of G is $\langle p_1, p_4, p_3 \rangle$, where p_i denotes the path in Π for hyperedge e_i of H. Intuitively, a line ordering at vertex u on the edge (u, u_i) is an extension of the embedding $\mathcal{G}$ to the ordering of the paths passing through (u, u_i). A *line ordering at vertex* u is the concatenation $\text{ord}_u(u_1) \oplus \cdots \oplus \text{ord}_u(u_k)$. A *vertex crossing* at u is a pair (p, q) of paths that appear in the line ordering at u alternatingly, that is, $p...q...p...q$; see the crossing between the green line and the orange line in vertex u in Fig. 2(a,b). A *line crossing* along an edge $e = (v, w)$ is a pair (p, q) of paths that appear in the line orderings at v and at w in the same order; refer to Fig. 2(c).

Minimizing the number of line crossings has been studied extensively in the context of drawing geographic metro maps [1,2,7,13,14]. We advocate that it is also interesting to minimize vertex crossings. Graphic designers sometimes seem to prefer them over line crossings; see Fig. 3.

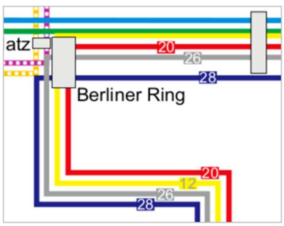


Fig. 3. A crossing vertex in a clipping of the official bus & tram map of Würzburg [18].

In this paper, we focus on vertex crossings and forbid line crossings. When representing abstract hypergraphs as metro maps, one has more freedom to place vertices; this can be used to produce drawings that avoid both types of crossings. For an example of such a drawing, see Fig. 4.

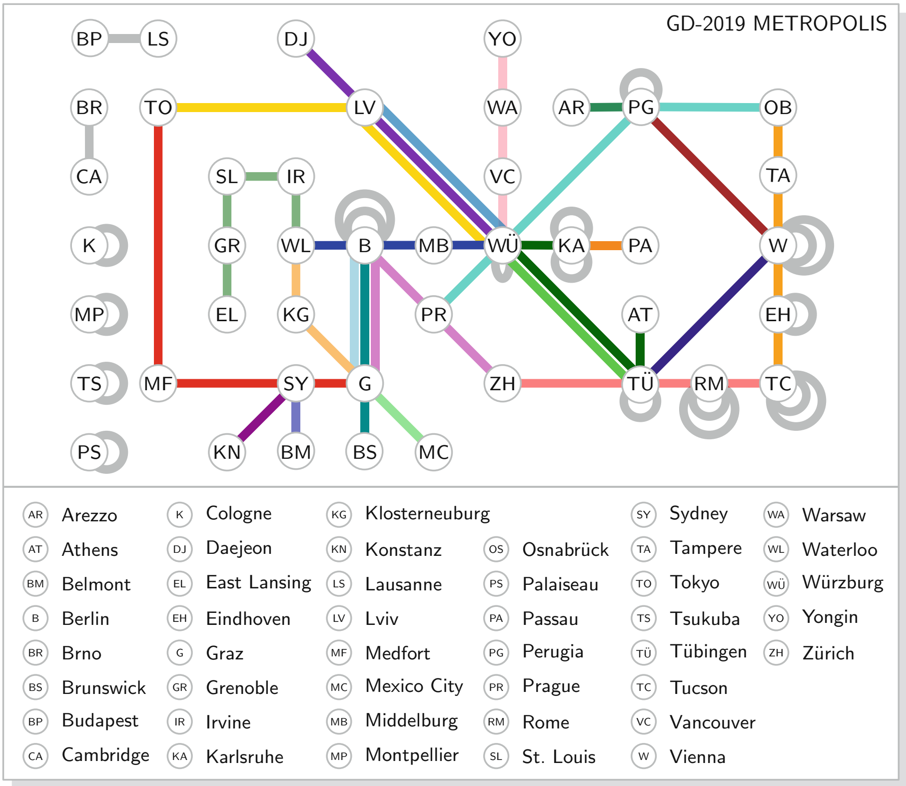


Fig. 4. A visualization of the conference GD 2019 as a metro map: the stations correspond to cities and the lines to papers. A line connects the cities where the authors of the corresponding paper are affiliated. The drawing has no vertex crossings.

We formalize our problem, which has two variants, as follows. In the *fixed-embedding* setting, an embedding $\mathcal{G}$ of G is given, and a *metro-map embedding* of $(\mathcal{G}, \Pi)$ is a set $\{\mathrm{ord}_u(v), \mathrm{ord}_v(u) \colon (u,v) \in E\}$ of line orderings. If the embedding of G is not part of the input—the *variable-embedding* setting—a *metro-map embedding* of (G, Π) is an embedding $\mathcal{G}$ of G and a metro-map embedding of $(\mathcal{G}, \Pi)$. We then define the problem Crossing Vertex Minimization: Given a pair (G, Π) or a pair $(\mathcal{G}, \Pi)$, we seek for a metro-map embedding that minimizes the *number of crossing vertices*, that is, the number of vertices containing vertex crossings—under the restriction that line crossings are not allowed.

Our contribution is as follows. We settle the complexity of Crossing Vertex Minimization, which turns out to be NP-hard in general, but polynomial-time solvable for trees. We also present an efficient algorithm for testing whether an instance $(\mathcal{G}, \Pi)$ or (G, Π) admits a metro-map embedding without any vertex crossings, for example as the one in Fig. 4. Table 1 gives an overview of our results.

We note that the problem of constructing a metro-map embedding in the fixed-embedding setting with a slightly different optimization goal was studied by Bast et al. [3]. The authors presented an ILP to minimize the total number of vertex crossings (as opposed to our optimization goal of minimizing the total number of crossing vertices).

Table 1. Our results for Crossing Vertex Minimization. Here k denotes the number of crossing vertices.

Problem type	Embedding	Graph	Result	Ref.
k Part of input	Fixed or Variable	Planar	NP-hard	Theorem 1
k Fixed	Fixed or Variable	Planar	Polynomial	Corollary 1
Optimization	Fixed or Variable	Tree	Polynomial	Theorem 3

2 Complexity

Theorem 1. Crossing Vertex Minimization *is NP-hard, both with fixed and variable embedding.*

Proof. We prove NP-hardness of the decision problem corresponding to Crossing Vertex Minimization by reducing from Planar Vertex Cover, which is defined as follows. Given a planar graph $G = (V, E)$ and a number k, is there a set S of k vertices such that, for every edge (u, v) in G, it holds that $u \in S$ or $v \in S$ (or both)?

Given an instance (G, k) of Planar Vertex Cover, we construct a planar graph $G' = (V', E')$ and a set Π' of paths in G' as follows (see Fig. 5):

$$\begin{aligned} V' &= V \cup \{x_e^1, x_e^2 \mid e \in E\} \\ E' &= E \cup \{(x_e^1, u), (x_e^2, u), (x_e^1, v), (x_e^2, v) \mid e = (u, v) \in E\} \\ \Pi' &= \{P_e^1 = [x_e^1, u, v, x_e^2], P_e^2 = [x_e^2, u, v, x_e^1] \mid e = (u, v) \in E\} \end{aligned}$$

In Fig. 5(b) and (c), vertices in V' are white, edges in E' are gray, and the two paths in Π' for the specific edge e are yellow and green. Clearly G' is planar.

We now claim that, for any embedding $\mathcal{G}'$ of G', there exists a metro-map embedding of $(\mathcal{G}', \Pi')$ with k crossing vertices if and only if G admits a vertex cover of size k. Note that this implies NP-hardness for both fixed *and* variable embeddings.

Given a metro-map embedding, for any edge $e = (u, v)$ in G, paths P_e^1 and P_e^2 necessarily cross, making at least one of u and v a crossing vertex. In other words, the set S of all crossing vertices forms a vertex cover of G.

Conversely, if S is a vertex cover of G, we can choose for each vertex $u \in V \setminus S$ and each edge $e = (u, v)$ incident to u in G the line ordering $\mathrm{ord}_u(v)$ such that P_e^1 and P_e^2 do not cross at u (but at v). For $i, j \in \{1, 2\}$ and for two different

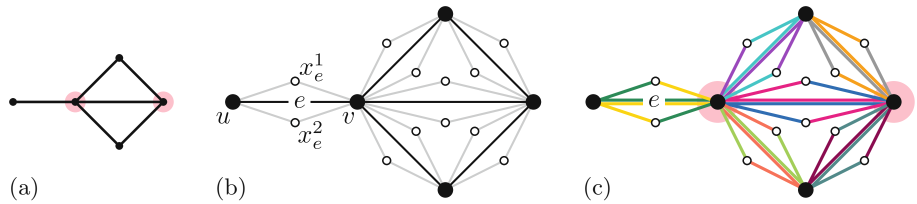


Fig. 5. Reduction from PLANAR VERTEX COVER to CROSSING VERTEX MINIMIZATION: (a) G with vertex cover (pink); (b) graph G'; (c) (G', Π) with two crossing vertices. (Color figure online)

edges $e' = (u, v')$ and $e'' = (u, v'')$ incident to u, paths $P^i_{e'}$ and $P^j_{e''}$ do not cross since, due to planarity, the triangles $[u, v', x^i_{e'}]$ and $[u, v'', x^j_{e''}]$ do not alternate along u. As crossings cannot occur at vertices in $V' \setminus V$, the resulting metro-map embedding has at most $|S| = k$ crossing vertices. □

3 Algorithms

We now turn to positive results, starting with metro-map embeddings without any vertex crossings. We formulate the corresponding decision problem as an instance of CLUSTERED PLANARITY, which was introduced by Lengauer [11] and independently by Feng et al. [6]. An instance of CLUSTERED PLANARITY consists of a planar graph H and a set $\mathcal{C}$ of subsets of vertices, called clusters. Any pair $C_1, C_2 \in \mathcal{C}$ of clusters is either disjoint or comparable by inclusion, i.e., $C_1 \cap C_2 \in \{\emptyset, C_1, C_2\}$. The task is to decide whether $(H, \mathcal{C})$ admits a clustered planar drawing, i.e., a crossing-free drawing of H together with a set of crossing-free closed Jordan curves, one for each cluster, such that each curve γ_C for cluster C contains exactly the vertices of C in its interior, and each curve crosses each edge at most once. Only very recently, Fulek and Tóth [9] showed that CLUSTERED PLANARITY can be decided efficiently. Their algorithm runs in $O(n^{16})$ time, where n is the number of vertices of the given planar graph. In the meantime, Bläsius et al. [4] came up with a simpler and faster algorithm, running in quadratic time.

For convenience, we denote the size of an instance (G, Π) for our metro-map embedding problems by $\|G, \Pi\|$ and remark that $\|G, \Pi\| = O(|V(G)| \cdot |\Pi|)$. While we state and prove the following theorem for the variable-embedding case, it is simple to adjust it to a given fixed embedding; see the discussion in Sect. 4.

Theorem 2. *Given a planar graph G and a set Π of paths in G, there is an algorithm that decides efficiently whether (G, Π) admits a metro-map embedding without vertex crossings. The algorithm runs in time $O(f(\|G, \Pi\|))$, where f denotes the time needed to decide an instance of* CLUSTERED PLANARITY.

Proof. Given (G, Π), we construct in $O(\|G, \Pi\|)$ time an equivalent instance $(H, \mathcal{C})$ of CLUSTERED PLANARITY. First, by adding single-edge paths to Π, we ensure that every edge of G lies in some path in Π. Then for each edge e of G we add a parallel[1] edge $\bar{e}$ to the graph and a new path $P(\bar{e})$ consisting just of $\bar{e}$ to the set of paths. The resulting instance $(G' = (V, E'), \Pi')$ admits a crossing-free metro-map embedding if and only if (G, Π) does.

For the graph H, we take as vertex set the incidences between vertices of G', edges of G', and paths in Π'. Formally, a vertex–edge incidence is a pair $(v, e) \in V \times E'$ with $v \in e$, a vertex–path incidence is a pair $(v, P) \in V \times \Pi'$ with $v \in V(P)$, and a vertex–edge–path incidence is a triple $(v, e, P) \in V \times E' \times \Pi'$ with $v \in e \in E(P)$. (Note that there are no more than $2|E'|$, $|V| \cdot |\Pi'|$, and $2|V| \cdot |\Pi'|$ instances of each type, respectively.) For each path $P = [v_1, \dots, v_p]$ in Π' with $v_i \in V$ $(i = 1, \dots, p)$, $e_i = v_i v_{i+1} \in E'$ $(i = 1, \dots, p-1)$ graph H contains a path $[(v_1, P), (v_1, e_1, P), (v_2, e_1, P), (v_2, e_2, P), \dots, (v_p, e_{p-1}, P), (v_p, P)]$ on all incidences of P as they appear along the path. We call these paths the *metro-line paths*. Additionally, for each edge $e = uv$ of the original graph G with parallel edge $\bar{e}$ in G', we put two paths $[(u, e), (u, \bar{e})]$ and $[(v, e), (v, \bar{e})]$ into H, which we simply call the *additional paths*. Thus H is the vertex-disjoint union of paths.

For the clustering, we first define, for each vertex–edge incidence (v, e), a cluster $C(v, e) = \{(v, e)\} \cup \{(v, e, P) \in V(H) \mid P \in \Pi'\}$. Second, for each vertex v in G', we define an inner cluster $C_{\text{in}}(v) = \{(v, P) \in V(H) \mid P \in \Pi'\}$ and an outer cluster $C_{\text{out}}(v) = C_{\text{in}}(v) \cup \bigcup_{(v,e) \in V(H)} C(v, e)$. Let $\mathcal{C}$ be the set of all these clusters. This completes the construction of the Clustered Planarity instance $(H, \mathcal{C})$. Clearly H is planar, and any pair of clusters in $\mathcal{C}$ is either disjoint or in inclusion-relation. Moreover we have that the size of $(H, \mathcal{C})$ is in $O(|V| \cdot |\Pi'|) = O(\|G, \Pi\|)$. See Fig. 6 for an illustration.

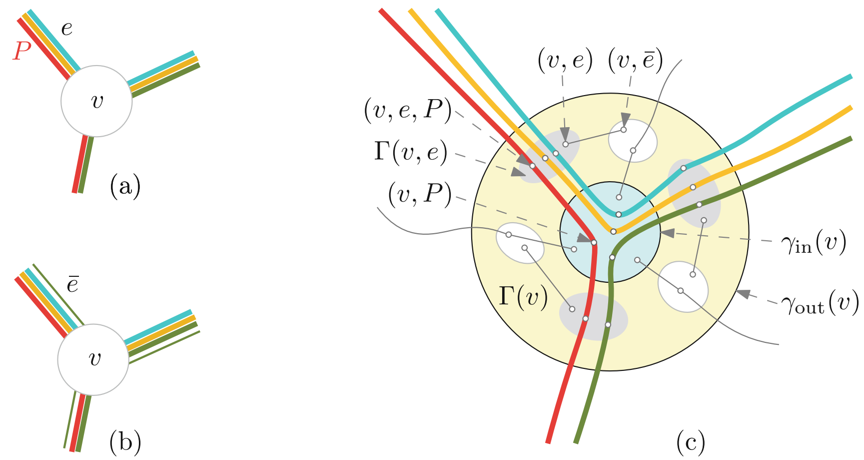


Fig. 6. Part of a crossing-free metro-map embedding of (G, Π) (a) and (G', Π') (b), and a corresponding clustered planar drawing of $(H, \mathcal{C})$ (c).

[1] To avoid parallel edges, paths of length two with the appropriate modifications would do equally well.

It remains to show that any clustered planar drawing of $(H, \mathcal{C})$ can be transformed into a metro-map embedding of the original instance (G, Π) without crossings. (The other direction is easy; see Fig. 6.)

For any vertex v in G, we have $C_{\text{in}}(v) \subsetneq C_{\text{out}}(v)$ and hence the corresponding closed curves $\gamma_{\text{out}}(v)$ and $\gamma_{\text{in}}(v)$ define an annulus-shaped region $\Gamma(v)$ (light yellow region in Fig. 6(c) in the plane that contains the region $\Gamma(v, e)$ (one of the gray regions in Fig. 6(c) for cluster $C(v, e)$ for every incident edge e at v. For every incidence (v, e, P), the metro-line path for P enters $\Gamma(v)$ from the outside, passes through $\Gamma(v, e)$, and leaves $\Gamma(v)$ to the inside. This gives a circular ordering σ_v of the incidence (v, e, P) around v. As $\Gamma(v, e) \cap \Gamma(v, e') = \emptyset$ for any e, e' at v, it follows that all incidences (v, e, P) for the same edge e appear consecutively in σ_v. Moreover, the additional path between (v, e) and $(v, \bar{e})$ implies that in σ_v the incidence for edge $\bar{e}$ appears next to the block of incidences for edge e.

We construct a crossing-free metro-map embedding of (G, Π) by drawing each vertex v inside its curve $\gamma_{\text{in}}(v)$, drawing each edge e of G along the metro-line path for $P(\bar{e})$ connecting the ends to v inside $\gamma_{\text{in}}(v)$ in a crossing-free way, and choosing the line ordering as the subordering of σ_v on incidences (v, e, P) with $P \in \Pi' - \Pi$. The constructed embedding of G is clearly crossing-free. Moreover, we have no vertex crossing at v as the metro-line paths do not cross inside $\gamma_{\text{in}}(v)$.

It remains to show that there are no line crossings, i.e., that for each edge $e = (u, v)$ in G, the line ordering at u on e is the reverse of the line ordering at v on e. (This would not be guaranteed without the parallel edges and additional paths, see Fig. 7.)

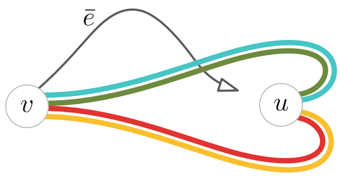


Fig. 7. Without the extra edge $\bar{e}$, a clustered planar drawing of $(H, \mathcal{C})$ might give line crossings if e is a bridge.

For the edge $e = (u, v)$, the circular ordering of incidences (v, e, P) around v is the reverse of the circular ordering of incidences (u, e, P) around u since the corresponding metro-line paths in H are non-crossing. Moreover, both sets of incidences appear as a consecutive block around the respective vertices. Finally, since the metro-line path for $\bar{e}$ starts and ends between (in the cyclic ordering at the vertices) the same metro-line paths for e at vertex u and vertex v, the line orderings $\text{ord}_u(v)$ and $\text{ord}_v(u)$ are indeed reversals of each other. In particular, a situation as shown in Fig. 7 is prevented. □

Theorem 2 implies the following.

Corollary 1. *For any fixed k, one can decide in polynomial time whether there is a metro-map embedding with at most k crossing vertices.*

Proof. We go through all $O(n^k)$ sets of k vertices for which we allow vertex crossings. Given such a set $S \subseteq V$, we split each path $P \in \Pi$ at every vertex of S. That is, consider Π_S to be the set of all inclusion-maximal subpaths of paths in Π with no inner vertex in S. By Theorem 2, we can test in polynomial time whether there is a metro-map embedding of (G, Π_S) without crossings. Clearly, such an embedding can be seen as a metro-map embedding of (G, Π) where all vertex crossings occur at vertices in S, and vice versa. As k is fixed, we obtain overall polynomial runtime. □

We now improve the result above for the case of trees.

Theorem 3. *There is an algorithm that solves* Crossing Vertex Minimization *for trees efficiently, both for variable and fixed embedding. Given a tree T and a set Π of paths, the algorithm runs in time $O(|V(T)| \cdot f(\|T, \Pi\|))$, where f denotes the time needed to decide an instance of* Clustered Planarity.

Proof. Let (T_0, Π_0) denote a given instance with T_0 being a tree and Π_0 a set of paths in T_0. We need to efficiently find a smallest subset S such that (T_0, Π_0) admits a metro-map embedding with every vertex crossing in S. Along with the set S, we obtain the edge-partition $\mathcal{T}(S)$ of T_0 into its inclusion-maximal subtrees with the property that each vertex of S is either a leaf in the subtree or not contained in it. To compute S, pick any vertex as the root in T_0 and process the vertices of the tree from the leaves towards the root, i.e., considering each vertex only if all its children have already been considered. On the way, we will remove vertices from tree, thereby removing or shortening some paths. We always denote by T the current tree and by Π the current set of paths.

Let T_v denote the subtree of T rooted at the currently processed vertex v and (if v is not the root) let w denote the parent of v. We consider the subtree T_v^+ of T on $T_v \cup w$, i.e., the tree T_v plus edge (w, v), and compute the set $\Pi|_{T_v^+} = \{P \cap T_v^+ \mid P \in \Pi\}$ of all paths in Π, each restricted to its maximal (possibly empty) subpath in T_v^+. We then use Theorem 2 to test in $O(f(\|T, \Pi\|))$ time whether $(T_v^+, \Pi|_{T_v^+})$ admits a metro-map embedding without any crossing. If it does, we consider v as successfully processed and continue with the next vertex. Otherwise, if $(T_v^+, \Pi|_{T_v^+})$ requires at least one vertex crossing, we add v to the set S and remove from T all vertices in T_v except for v, i.e., continue with the instance $(T_v^- = T - (T_v - v), \Pi|_{T_v^-})$. Observe that if $u_1, \ldots, u_k$ are the children of v, then T is the edge-disjoint union of $T_{u_1}^+, \ldots, T_{u_k}^+$ and T_v^-.

Once every vertex of T_0 is processed, we have computed a set $S \subseteq V(T_0)$ with corresponding edge-partition $\mathcal{T}(S)$ with the properties that

(i) for every tree $T' \in \mathcal{T}(S)$ the instance $(T', \Pi_0|_{T'})$ admits a crossing-free metro-map embedding, and

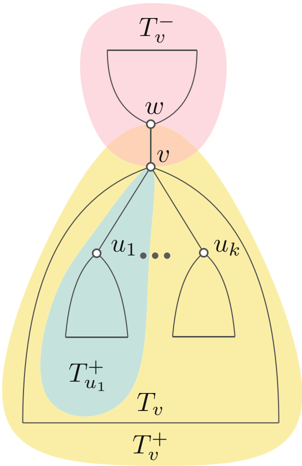


Fig. 8. Illustration of the subtrees of T for v and its children $u_1, \dots, u_k$. Note that T is the edge-disjoint union of $T_v^- = T - (T_v - v)$ and $T_{u_1}^+, \dots, T_{u_k}^+$.

(ii) if $T_1, \dots, T_k \in \mathcal{T}(S)$ are the trees containing the edges of a vertex $v \in S$ with parent w (if existent) to its k children, then every metro-map embedding of $(T_v^+ = T_1 \cup \dots \cup T_k \cup \{vw\}, \Pi_0|_{T_v^+})$ has at least one vertex crossing.

Combining the metro-map embeddings of all trees in $\mathcal{T}(S)$ given by (i) gives a metro-map embedding of (T_0, Π_0) with all vertex crossings in S, i.e., $|S|$ crossing vertices are sufficient. On the other hand, (ii) implies that every metro-map embedding of T_0 has at least one vertex crossing at a non-leaf vertex of T_v^+. As for distinct $u, v \in S$, trees T_v^+ and T_u^+ do not share non-leaf vertices, $|S|$ crossing vertices are necessary.

The runtime of our algorithm is in $O(|V(T)| \cdot f(\|T, \Pi\|))$ since it is dominated by the $|V(T)|$ calls to Theorem 2. □

4 Discussion

First, note that the algorithm in Theorem 2, and hence also those in Corollary 1 and Theorem 3, assume that G has variable embedding. Of course, we can handle the fixed-embedding setting by triangulating the given embedded graph in a preprocessing step. This also holds for the treatment of trees in Theorem 3 by doing a separate triangulation for each call of Theorem 2 therein.

Second, we leave as an open problem whether Crossing Vertex Minimization is fixed-parameter tractable.

Third, refer back to Fig. 1. Intuitively, the metro-map drawing on the right could be improved by switching the order of paths p_1 and p_4 on edge $(1, 4)$. However, this improvement is not reflected by objective functions considered so far since the drawing has neither vertex nor line crossings. Maybe there is a need for a more fine-grained objective?

Acknowledgments. We thank the organizers and the other participants of the 2017 Dagstuhl seminar "Scalable Set Visualization", where this work started, in particular Robert Baker, Nan Cao, and Yifan Hu.

References

1. Argyriou, E., Bekos, M.A., Kaufmann, M., Symvonis, A.: On metro-line crossing minimization. J. Graph Algorithms Appl. **14**(1), 75–96 (2010). https://doi.org/10.7155/jgaa.00199
2. Asquith, M., Gudmundsson, J., Merrick, D.: An ILP for the metro-line crossing problem. In: Proceedings 14th Computing Australasian Theory Symposium (CATS 2008), vol. 77, pp. 49–56. CRPIT (2008). https://crpit.scem.westernsydney.edu.au/confpapers/CRPITV77Asquith.pdf
3. Bast, H., Brosi, P., Storandt., S.: Efficient generation of geographically accurate transit maps. In: Proceedings 26th ACM SIGSPATIAL International Conference on Advances in Geographic Information Systems (SIGSPATIAL 2018), pp. 13–22 (2018). https://doi.org/10.1145/3274895.3274955
4. Bläsius, T., Fink, S.D., Rutter, I.: Synchronized planarity with applications to constrained planarity problems. arXiv (2020). https://arxiv.org/abs/2007.15362
5. Brandes, U., Cornelsen, S., Pampel, B., Sallaberry, A.: Path-based supports for hypergraphs. J. Discrete Algorithms **14**, 248–261 (2012). https://doi.org/10.1016/j.jda.2011.12.009
6. Feng, Q.-W., Cohen, R.F., Eades, P.: Planarity for clustered graphs. In: Spirakis, P. (ed.) ESA 1995. LNCS, vol. 979, pp. 213–226. Springer, Heidelberg (1995). https://doi.org/10.1007/3-540-60313-1_145
7. Fink, M., Pupyrev, S., Wolff, A.: Ordering metro lines by block crossings. J. Graph Algorithms Appl. **19**(1), 111–153 (2015). https://doi.org/10.7155/jgaa.00351
8. Foo, B.: The memory underground. http://memoryunderground.com
9. Fulek, R., Tóth, C.D.: Atomic embeddability, clustered planarity, and thickenability. In: Proceedings ACM-SIAM Symposium Discrete Algorithms (SODA), pp. 2876–2895 (2020). https://doi.org/10.1137/1.9781611975994.175
10. Honnorat, D.: The best movies of all time map (2009). http://www.dyblog.fr/index.php?2009/07/20/928-une-carte-des-250-meilleurs-films-de-tous-les-temps+www.dyblog.fr/index.php?2009/07/20/928-une-carte-des-250-meilleurs-films-de-tous-les-temps+, Accessed 8 Jun 2020
11. Lengauer, T.: Hierarchical planarity testing algorithms. J. ACM **36**(3), 474–509 (1989). https://doi.org/10.1145/65950.65952
12. Nesbitt, K.V.: Getting to more abstract places using the metro map metaphor. In: Proceedings 8th International Conference Information Vision (IV 2004), pp. 488–493. IEEE (2004). https://doi.org/10.1109/IV.2004.1320189
13. Nöllenburg, M.: An improved algorithm for the metro-line crossing minimization problem. In: Eppstein, D., Gansner, E.R. (eds.) GD 2009. LNCS, vol. 5849, pp. 381–392. Springer, Heidelberg (2010). https://doi.org/10.1007/978-3-642-11805-0_36
14. Nöllenburg, M., Wolff, A.: Drawing and labeling high-quality metro maps by mixed-integer programming. IEEE Trans. Vis. Comput. Graph. **17**(5), 626–641 (2011). https://doi.org/10.1109/TVCG.2010.81

15. Sandvad, E., Grønbæk, K., Sloth, L., Knudsen, J.L.: A metro map metaphor for guided tours on the Web: the Webvise guided tour system. In: Shen, V.Y., Saito, N., Lyu, M.R., Zurko, M.E. (eds.) Proceedings 10th International World Wide Web Conference (WWW 2001), pp. 326–333. ACM (2001). https://doi.org/10.1145/371920.372079
16. Shahaf, D., Yang, J., Suen, C., Jacobs, J., Wang, H., Leskovec, J.: Information cartography: Creating zoomable, large-scale maps of information. In: Proceedings 19th ACM SIGKDD Conference. Knowledge Discovery and Data Mining (KDD), pp. 1097–1105 (2013). https://doi.org/10.1145/2487575.2487690
17. Stott, J.M., Rodgers, P., Burkhard, R.A., Meier, M., Smis, M.T.J.: Automatic layout of project plans using a metro map metaphor. In: Proceedings 9th International Conference Information Vision (IV 2005), pp. 203–206. IEEE (2005). https://doi.org/10.1109/IV.2005.26
18. Verkehrsunternehmensverbund Mainfranken. Liniennetz Würzburger Straenbahn (2019). https://www.wvv.de/de/media/downloads/downloadcenter/hauptnavigation/mobilitaet/liniennetz/2019-11-01-liniennetzplan-busse-und-strabas_a3.pdf, Accessed 8 Jun 2020