# Short plane support trees for hypergraphs

# Short Plane Support Trees for Hypergraphs*

**Thom Castermans[1], Mereke van Garderen[2], Wouter Meulemans[1], Martin Nöllenburg[3], and Xiaoru Yuan[4]**

1 **Dept. of Mathematics and Computer Science, TU Eindhoven, the Netherlands**
[t.h.a.castermans|w.meulemans]@tue.nl
2 **Dept. of Computer and Information Sciences, Universität Konstanz, Germany**
mereke.van.garderen@uni-konstanz.de
3 **Algorithms and Complexity Group, TU Wien, Vienna, Austria**
noellenburg@ac.tuwien.ac.at
4 **Peking University, Beijing, China**
xiaoru.yuan@pku.edu.cn

## Abstract

In many domains, the aggregation or classification of data elements leads to various intersecting sets. To allow for intuitive exploration and analysis of such data, set visualization aims to represent the elements and sets graphically. In more theoretical literature, such set systems are often referred to as hypergraphs. A support graph is a notion for drawing such a hypergraph, understood as a regular graph spanning the same vertices (elements), in which each hyperedge (set) induces a connected subgraph.

In this paper, we investigate finding a support graph of a hypergraph with fixed vertex locations under various constraints. We focus on enforcing planarity using a straight-line embedding, while minimizing the total length of the edges of the support graph, and consider the effect of the additional requirement that the support graph is acyclic.

## 1 Introduction

Intersecting sets are used in many domains to model various ways of clustering, grouping or aggregating measurements or data elements. To allow for effective exploration and analysis of such set systems, visualization is often used. Indeed, set visualization is an active subfield of information visualization; Alsallakh *et al.* [3] recently surveyed it. We focus on the case where elements have fixed positions in the plane, arising e.g. from geospatial locations.

On the theoretical side, such a set system is often referred to as a *hypergraph* $H = (V, S)$, with a set of vertices $V$ (elements) and hyperedges $S$ (sets), where each hyperedge $s \in S$ is some nonempty subset of $V$. A *support graph* of a hypergraph $H = (V, S)$ is a (regular) graph $G = (V, E)$ on the same vertex set such that every hyperedge $s \in S$ induces a connected subgraph in $G$ [8]. In the remainder, we assume that $V$ is a set of points in the plane and that a support graph is embedded using straight-line edges.

Though there are various ways of visualizing sets, support graphs match to a popular style in set visualization, namely that of connecting elements using colored links, such as seen for example in Kelp-style diagrams [9, 13] (see also Fig. 1) or LineSets [2]. Finding an embedded support graph that satisfies certain criteria therefore readily translates into a good rendering of the corresponding set system. A "good" support graph should avoid edge crossings, a standard quality criterion in the graph-drawing literature [14]. Moreover, as per Tufte's principle of ink minimization [15], it should have small total edge length.

* This work was started at Dagstuhl seminar 17332, Scalable Set Visualizations. T. Castermans is supported by the Netherlands Organisation for Scientific Research (NWO, 314.99.117). W. Meulemans is partially supported by the Netherlands eScience Centre (NLeSC, 027.015.G02).

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

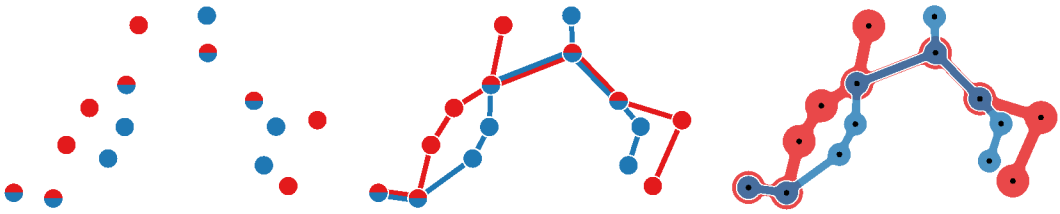

**Figure 1** (a) A set system with colors indicating set membership. (b) The shortest plane support of the corresponding hypergraph. (c) A Kelp-style rendering of the set system.

**Contributions** In Section 2, we explore theoretical properties of requiring planarity. In particular, if there is at least one vertex that occurs in all hyperedges, then a plane support tree exists. However, we show that the minimum spanning tree on these vertices is not necessarily contained in any support tree or graph that is an approximation of the shortest one. An analogous statement holds for plane versus nonplane support trees. In Section 3, we turn to computational aspects and show that finding a plane support tree or graph with minimal total edge length is NP-hard, even for only two hyperedges that are disjoint or if one contains the other. In Section 4, we sketch an integer linear program to solve the problem.

**Related Work** Regarding support graphs for elements with fixed locations, some results are already known. The results of Bereg *et al.* [5] imply that existence of a plane support tree for two disjoint hyperedges can be decided in polynomial time; this readily implies the same result for a plane support graph. To the best of our knowledge, the problem is still open for $|S| > 2$; our result proves a sufficient condition but not a necessary one. This problem has also been studied in a Steiner setting [4], where additional points may be placed. Van Goethem *et al.* [16] enforce a stricter planarity than that of planar supports and investigate the resulting properties for elements on a regular grid, where only neighboring elements can be connected. However, length of the solution is of no concern in their results.

Without planarity, existence and length minimization of a (nonplane) support tree for fixed elements can be solved in polynomial time [11, 12]. However, acyclicity makes this problem easier. Indeed, length minimization of a (nonplane) support graph is NP-hard for three or more hyperedges [1]. In contrast, we show that this is in fact hard for two hyperedges if we require a plane support graph or tree. Without the planarity requirement, Hurtado *et al.* [10] show that length minimization for two hyperedges is solvable in polynomial time.

Planar support graphs without fixed elements have also received attention. For example, Buchin *et al.* [8] show that deciding whether a planar support graph exists is NP-hard in general; this proof readily works for elements with fixed locations, but note that it requires many hyperedges. In contrast, our result requires only two hyperedges, but uses length minimization. Brandes *et al.* [7] investigate support trees without fixed vertex locations, under the additional constraint that the induced subgraph of each hyperedge is Hamiltonian, and show that the existence of such a support can be checked in polynomial time.

## 2 Existential results

Before we study the computational problem of finding shortest plane support trees for a given hypergraph, we observe that a plane support tree always exists if there is at least one vertex that is contained in all hyperedges. To see why this is the case, consider the minimum spanning tree $T$ of the nonempty set $A = \bigcap_{s \in S} s$, which is crossing-free. We can connect

each remaining point to its closest point in $A$. The resulting graph is obviously a tree since we add only leaves to $T$, it is a support as every hyperedge induces a connected subgraph, and it is plane as no edge crossings are created when connecting to the closest point in $A$.

▶ Observation 1. Consider a hypergraph $H = (V, S)$ with no three vertices in $V$ on a line, such that $\bigcap_{s \in S} s \neq \emptyset$. $H$ has a plane support tree.

We note that in a support tree the subgraph induced by $A$ must be a connected subtree in order to satisfy the support property for all hyperedges. Next we show that using the above idea to start with a minimum spanning tree of $A$, the length of the resulting support tree cannot be bounded to be within a constant factor of the shortest plane support tree.

▶ **Lemma 2.1.** *There is a family of $n$-vertex hypergraphs $H = (V, S)$ with two hyperedges $S = \{r, b\}$ and $A = r \cap b \neq \emptyset$ such that any plane support tree of $H$ that includes a minimum spanning tree of $A$ is a factor $O(n)$ longer than the shortest plane support tree.*

**Proof.** The hypergraph family is illustrated in Fig. 2. The set $A = \{u, v, w\}$ consists of three vertices whose minimum spanning tree $T$ has length $\ell + 1$ and is indicated by the black edges in Fig. 2(a). The remaining vertices in $V \setminus A$ are indicated in red and blue (indicating membership of $r$ and $b$) and placed inside a disk of radius $\varepsilon$ just left of the midpoint of edge $uv$. The vertices alternate in colors from left to right and form two mirrored convex chains.

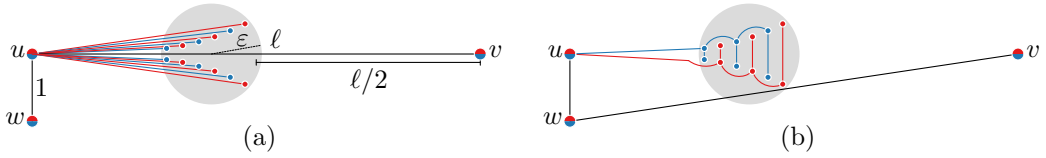


**Figure 2** An $n$-point instance with approximation ratio $O(n)$ if using a minimum spanning tree on $A$. All edges are straight-line segments; curvature just emphasizes the effect of the convex chain.

Since edge $uv$ of $T$ splits the vertices in $V \setminus A$ and by their placement on convex chains, the shortest extension of $T$ into a plane support tree is to connect every vertex to $u$ (Fig. 2(a)). This yields a total length of the support tree of $O(n) \cdot \ell$. If, however, $A$ is connected by a slightly longer tree, the remaining vertices in $V \setminus A$ can be joined by two comb-shaped structures as shown in Fig. 2(b). The resulting plane support tree has length of $O(1) \cdot \ell$. ◀

The above result also holds if we allow a general plane support graph. By removing the vertex $w$ from the instance of Fig. 2 one can show in a similar fashion that a plane support tree, which now necessarily includes the edge $uv$, is a factor $O(n)$ longer than a shortest nonplane support tree; this corollary does not immediately generalize to support graphs.

▶ **Corollary 2.2.** *There is a family of $n$-vertex hypergraphs $H = (V, S)$ with two hyperedges $S = \{r, b\}$ and $A = r \cap b \neq \emptyset$ such that any plane support tree of $H$ is a factor $O(n)$ longer than the shortest nonplane support tree.*

## 3 Computing a shortest plane support graph is NP-hard

Let us now turn towards the computational problem of finding the shortest plane support graph. Unfortunately, this problem and several restricted variants are NP-hard.

▶ **Theorem 3.1.** *Let $H = (V, S)$ be a hypergraph with vertices $V$ having fixed locations in $\mathbb{R}^2$ and with $S$ containing two hyperedges $r$ and $b$ such that $r \subseteq b$. It is NP-hard to decide whether $H$ admits a plane support tree with length at most $L$ for some $L > 0$.*

**Proof.** We use a reduction from (rectilinear) planar monotone 3-SAT [6]. Here, we are given a 3-CNF formula $\phi$ with $n$ variables $v_1, \ldots, v_n$ and $m$ clauses $c_1, \ldots, c_m$ such that every clause either has three positive literals or three negative literals. Moreover, we are given an embedding of $\phi$ as a plane graph, with rectangular vertices for variables on a horizontal line, and clauses as rectangles above or below the line (depending on whether the clause is positive or negative). Vertical edges connect clauses to the variables of their literals. We assume without loss of generality that the clauses are numbered according to their nesting: that is, $c_i < c_j$ if $c_i$ is closer to the line of vertices than $c_j$ in the embedding.

We must construct a hypergraph $H = (V, \{r, b\})$ such that $r \subseteq b$. In the remainder, we assign vertices to either $r$ (red) or $b$ (blue), understanding that any red vertex is also in $b$.

First, we place $3(n+1)$ red vertices using coordinates $(3i \cdot (m+1), y)$ for integers $i \in [0, n]$ and integers $y \in [-1, 1]$. Furthermore, we place $n \cdot (3m + 2)$ blue vertices using coordinates $(3i(m + 1) + j, 0)$ for integers $i \in [0, n - 1]$ and $j \in [1, 3m + 2]$.

We now place additional blue vertices for each clause $c_a$. We assume that this clause has positive literals for variable $v_i$, $v_j$, and $v_k$; the construction for clauses with negative literals is symmetric, using negative $y$-coordinates instead. First, we place $3a + 1$ blue vertices from $(3(i - 1)(m + 1) + 3p, 2)$ to $(3(i - 1)(m + 1) + 3p, 2 + 3a)$ at unit distance, to represent the incidence from $c_a$ to variable $v_i$, using the given embedding to determine that $c_a$ is the $p$th clause incident from above to $v_i$. Analogously, we place the blue vertices for $v_j$ and $v_k$. Now, we place further blue vertices at unit distance with $y$-coordinate $2 + 3a$ from the leftmost to the rightmost top vertex we just placed. The result is given in Fig. 3.

One clause requires at most $3(3m + 1)$ vertices for the variable incidence and less than $3n \cdot (m + 1)$ for the horizontal line connecting these. We can now readily measure the length of the minimum spanning tree on the blue vertices of one clause. We use $L_a$ to denote this length; note that $L_a$ is an integer at most $3(3m + 1) + 3n \cdot (m + 1)$.

The value of $L$ that we select is $2(n + 1) + 3n \cdot (m + 1) + n(3m + 2) + 2m + \sum_{a \in [1,m]} L_a$.

This finalizes the construction. It is polynomial since we placed $3(n + 1)$ red vertices and $n \cdot (3m - 2)$ blue vertices for the variables and at most $m \cdot (3(3m + 1) + 3n \cdot (m + 1))$ for the clauses: this is $O(nm^2)$ vertices. Moreover, we claim that our constructed hypergraph admits a plane support tree of length at most $L$, if and only if $\phi$ is satisfiable.

Assume we have a plane support tree of length at most $L$. First, we observe that all

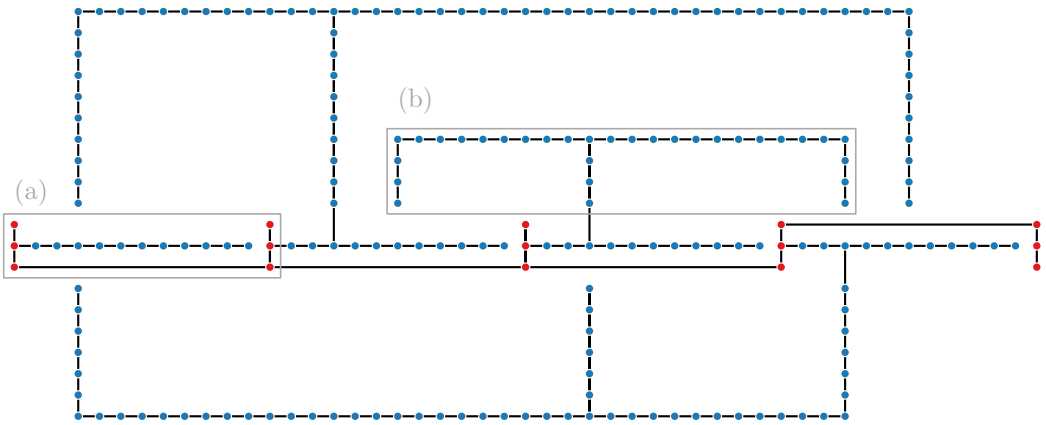


**Figure 3** Construction for $\phi = (v_2 \vee v_3 \vee v_4) \wedge (\overline{v_1} \vee \overline{v_3} \vee \overline{v_4}) \wedge (v_1 \vee v_2 \vee v_4)$. Vertices in $r$ and $b$ are red, vertices in $b$ are blue. A plane support tree with length at most $L$ is given in black lines. (a) Representation of variable $v_1$; the solution sets $v_1$ to true. (b) Representation of the first clause.

points in $r$ must be connected: the minimal way of doing so connects the three vertices with the same $x$-coordinate and uses one horizontal line to connect one triplet to the next. This has exactly length $2(n+1)+3n\cdot(m+1)$, corresponding to the first two terms defining $L$. The minimal way of connecting the lines inside the variables to the red tree takes length $n(3m+2)$ in total: this is the third term defining $L$. Finally, to connect the clause vertices, we need length at least $L_a$ per clause, the last term of $L$. We note that any solution must use these constructions on the blue vertices, since all vertices are at unit distance; other blue vertices are at distance at least 2. However, the support tree is connected: thus it must still have connections from each gadget to either a red vertex or a blue vertex of a variable. The budget we have for this is $2m$ in total. Since each clause needs a connection of length at least 2, all clauses use exactly length 2. The only vertices within distance 2 of a clause are the three blue vertices of the variables with $y$-coordinate zero (one of each literal of the clause). Thus, each clause must have exactly one length-2 edge to one of these variable vertices. Since the support tree is plane, this cannot cross the horizontal links used to connect the red vertices. We can now readily obtain a satisfying assignment for $\phi$, by looking at which of the two horizontal lines is used to connect the red vertices: if the one at the top is used, that variable is set to false; it is set to true otherwise.

To prove the converse, assume that we have a satisfying assignment. Using the same reasoning as above, we can construct the plane support tree by picking the connecting horizontal lines for the red vertices according to the satisfying assignment: this readily leads us to conclude that we can connect each clause using a length-2 connection that does not intersect the horizontal lines for the red vertices. ◀

We observe that the above proof readily implies that finding the shortest plane support graph is also NP-hard, as is the case that $r$ is not a subset of $b$. Moreover, the proof can be easily adapted to show the other special case of disjoint $r$ and $b$: this needs slightly more spacing such that we can add a few extra blue vertices that can be used to connect all the blue vertices of the variables into a single component using only length-1 edges.

## 4 Integer linear program

We showed in Section 3 that finding the shortest plane support is NP-hard, and so are several restricted versions of that problem. It is however possible to formulate these problems as *integer linear programs* (ILP), allowing us to leverage effective ILP solvers. Below, we briefly sketch how to obtain an ILP for a hypergraph $H=(V,S)$.

We introduce variables $e_{u,v}\in\{0,1\}$, indicating whether edge $uv$ is selected for the support graph. This readily allows us to represent a graph with fixed vertices. Because the vertex locations are fixed, we can precompute edge lengths $d_{u,v}$ as well as which pairs of edges intersect. This gives the following basic program

$$\begin{aligned} \text{minimize} \quad & \textstyle\sum_{u,v\in V} d_{u,v}\cdot e_{u,v} \\ \text{subject to} \quad & e_{u,v}+e_{w,x}\le 1 \qquad \text{for all } u,v,w,x\in V \text{ if edges } uv \text{ and } wx \text{ intersect.} \end{aligned}$$

What remains is to ensure that the graph is also a support: we need additional constraints that imply that each hyperedge in $S$ induces a connected subgraph. To this end, we construct a flow tree for each hyperedge $s$. We pick an arbitrary sink for the hyperedge, $\sigma_s\in s$, that may receive flow, and let the remaining vertices in $s$ generate one unit of flow. To formalize this, we introduce variables $f_{s,u,v}\in\{0,1,\dots,|s|-1\}$ for each $s\in S$ and $u,v\in s$ with $u\neq v$. We now need the following constraints: (a) the incoming flow at $\sigma_s$ is exactly $|s|-1$; (b)

the outgoing flow at $\sigma_s$ is zero; (c) except for $\sigma_s$, each vertex in $s$ sends out one unit of flow more than it receives; (d) flow can be sent only over selected edges.

$$\begin{array}{lll} \text{(a)} & \sum_{u \in s \setminus \{\sigma_s\}} f_{s,u,\sigma_s} = |s| - 1 & \text{for all } s \in S \\ \text{(b)} & f_{s,\sigma_s,v} = 0 & \text{for all } s \in S, v \in s \setminus \{\sigma_s\} \\ \text{(c)} & \sum_{v \in s \setminus \{u\}} (f_{s,u,v} - f_{s,v,u}) = 1 & \text{for all } s \in S, u \in s \setminus \{\sigma_s\} \\ \text{(d)} & f_{s,u,v} \leq e_{u,v} \cdot (|s| - 1) & \text{for all } s \in S, u, v \in s \text{ with } u \neq v \end{array}$$

**Variants** The above ILP results in the shortest plane support graph for $H$. It can easily be modified to give a (shortest plane) support tree as well as to penalize or admit a limited number of intersections. The latter requires additional variables to indicate whether both edges of a crossing pair are used.

## References


1 H. A. Akitaya, M. Löffler, and C. D. Tóth. Multi-colored spanning graphs. In *Graph Drawing and Network Visualization (GD'16)*, LNCS 9801, pp. 81–93. Springer, 2016.

2 B. Alper, N. Henry Riche, G. Ramos, and M. Czerwinski. Design study of LineSets, a novel set visualization technique. *IEEE TVCG*, 17(12):2259–2267, 2011.

3 B. Alsallakh, L. Micallef, W. Aigner, H. Hauser, S. Miksch, and P. Rodgers. The state of the art of set visualization. *CGF*, 35(1):234–260, 2016.

4 S. Bereg, K. Fleszar, P. Kindermann, S. Pupyrev, J. Spoerhase, and A. Wolff. Colored non-crossing Euclidean Steiner forest. In *Algorithms and Computation (ISAAC'15)*, LNCS 9472, pp. 429–441. Springer, 2015.

5 S. Bereg, M. Jiang, B. Yang, and B. Zhu. On the red/blue spanning tree problem. *TCS*, 412(23):2459–2467, 2011.

6 M. de Berg and A. Khosravi. Optimal binary space partitions in the plane. In *Computing and Combinatorics (COCOON'10)*, LNCS 6196, pp. 216–225. Springer, 2010.

7 U. Brandes, S. Cornelsen, B. Pampel, and A. Sallaberry. Path-based supports for hypergraphs. In *Combinatorial Algorithms (IWOCA'10)*, LNCS 6460, pp. 20–33. Springer, 2010.

8 K. Buchin, M. van Kreveld, H. Meijer, B. Speckmann, and K. Verbeek. On planar supports for hypergraphs. *JGAA*, 14(4):533–549, 2011.

9 K. Dinkla, M. van Kreveld, B. Speckmann, and M. Westenberg. Kelp Diagrams: Point set membership visualization. *CGF*, 31(3pt1):875–884, 2012.

10 F. Hurtado, M. Korman, M. van Kreveld, M. Löffler, V. Sacristán, A. Shioura, R. I. Silveira, B. Speckmann, and T. Tokuyama. Colored spanning graphs for set visualization. *CGTA*, 68:262–276, 2018.

11 B. Klemz, T. Mchedlidze, and M. Nöllenburg. Minimum tree supports for hypergraphs and low-concurrency Euler diagrams. In *Algorithm Theory (SWAT'14)*, LNCS 8503, pp. 253–264. Springer, 2014.

12 E. Korach and M. Stern. The clustering matroid and the optimal clustering tree. *Mathematical Programming, Series B*, 98:385–414, 2003.

13 W. Meulemans, N. Henry Riche, B. Speckmann, B. Alper, and T. Dwyer. KelpFusion: A hybrid set visualization technique. *IEEE TVCG*, 19(11):1846–1858, 2013.

14 H. Purchase. Metrics for graph drawing aesthetics. *J Visual Languages & Computing*, 13(5):501–516, 2002.

15 E. Tufte. *The Visual Display of Quantitative Information*. Graphics Press, 2001.

16 A. van Goethem, I. Kostitsyna, M. van Kreveld, W. Meulemans, M. Sondag, and J. Wulms. The painter's problem: covering a grid with colored connected polygons. In *Graph Drawing and Network Visualization (GD'17)*, 2017. `arXiv:1709.00001`.