

Improving Temporal Treemaps by Minimizing Crossings: Supplementary Material

Alexander Dobler, Martin Nöllenburg

April 4, 2024

1 Introduction

This appendix to the paper *Improving Temporal Treemaps by Minimizing Crossings* contains two main sections. Section 2 gives an algorithm to decide if a temporal tree has a combinatorial layout with zero crossings, i.e., a planar combinatorial layout. Section 3 extends upon the application example given in the paper and shows multiple layouts computed by the different algorithms. This should show how different numbers of crossings impact the visualizations as temporal treemaps of the different temporal trees.

2 Recognizing Planar Instances

We know that we cannot find a polynomial-time algorithm that computes combinatorial layouts with the minimum number of (leaf) crossings for every temporal tree (assuming $P \neq NP$) because of the NP-hardness shown in the paper. However, we can decide whether a hierarchy-compliant temporal tree has a combinatorial layout with zero (leaf) crossings. We call such temporal trees *planar*. Köpp and Weinkauff [KW19] left this as an open problem. First, notice the following.

Theorem 2.1. *A combinatorial layout of a hierarchy-compliant temporal tree \mathcal{L} has zero crossings if and only if it has zero leaf crossings.*

Proof. The forward direction is immediate as every leaf crossing is a crossing. Assume \mathcal{L} has zero leaf crossings. We show that \mathcal{L} has zero crossings as well. Assume to the contrary that \mathcal{L} has a crossing between edges $e = (u, v)$ and $f = (u', v')$. Then there exist leaf edges $g = (a, b)$ and $h = (a', b')$ such that a, b, a', b' are descendants of u, v, u', v' , respectively. Thus, g and h form a leaf crossing, giving a contradiction. \square

We now give a polynomial-time algorithm to decide whether a hierarchy-compliant temporal tree \mathcal{G} is planar. We apply a linear-time algorithm for the problem of \mathcal{T} -level planarity testing [ALB⁺15, JLM98]. In this problem we are given as input a sequence of trees $\mathcal{T} = (T_1, T_2, \dots, T_\ell)$ such that there exist additional edges E between the leaves of consecutive trees T_i and T_{i+1} , for $i = 1, \dots, \ell - 1$. The question is if \mathcal{T} is *planar*, i.e., if there exist permutations $(\pi_1, \pi_2, \dots, \pi_\ell)$ such that

- π_i is a permutation of the leaves of T_i , $i = 1, 2, \dots, \ell$,
- the leaves in $T_i(u)$ for an internal node u of T_i appear consecutively in π_i , and
- there are no two edges $e = (u, v)$ and $f = (u', v')$ between trees T_i and T_{i+1} crossing each other, i.e., $u \prec_{\pi_i} u'$ and $v' \prec_{\pi_{i+1}} v$, or $u' \prec_{\pi_i} u$ and $v \prec_{\pi_{i+1}} v'$ never happens.

Contrary to temporal trees, here, edges only connect leaves of consecutive trees. In what follows we assume that every leaf of any tree $\mathcal{G}|_{t_i}$ is the endpoint of at least one temporal edge. Otherwise, such a leaf can never contribute crossings and can simply be removed for the purposes of the following description. We can now transform a temporal tree \mathcal{G} into such a sequence of trees \mathcal{T} , such that \mathcal{T} is planar if and only if \mathcal{G} is planar. Namely, let V_L be the set of all nodes in \mathcal{G} that are incident to at least one leaf edge. Note that all leaves are in this set as we required each leaf to have at least one

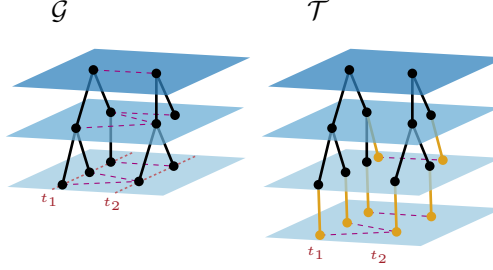


Figure 1: Illustration of the construction of \mathcal{T} from \mathcal{G} . The newly added nodes and hierarchical edges are drawn in orange. Note that edges in \mathcal{T} always connect leaves of the corresponding trees.

incident temporal edge. For each $u \in V_L$ add a new leaf node u' and connect u to u' . That is, u' is now the leaf of u in the corresponding tree. Now for each leaf edge (u, v) in \mathcal{G} between t_i and t_{i+1} , we add the edge (u', v') for the newly constructed leaf children u' of u and v' of v . Lastly, we remove all temporal edges from \mathcal{G} such that at least one of its endpoints is not a leaf. This construction is illustrated in Fig. 1 for a temporal tree \mathcal{G} on two timesteps t_1 and t_2 . Consider an arbitrary temporal tree \mathcal{G} and a sequence of trees \mathcal{T} obtained by the above construction.

Theorem 2.2. \mathcal{G} is planar if and only if \mathcal{T} is planar.

Proof. “ \Rightarrow ”: Let \mathcal{L} be a combinatorial layout $(\pi_1, \pi_2, \dots, \pi_\ell)$ of \mathcal{G} without leaf crossings. Consider two leaves u', v' of tree T_i for $i \in [\ell]$. Let u and v be the parents of u' and v' in T_i , respectively. Notice that u and v also appear in $\mathcal{G}|_{t_i}$. We construct the permutation π'_i of leaves in T_i such that $u' \prec_{\pi'_i} v'$ if and only if $u \prec_{\pi_i} v$ for all such pairs u, v of leaves (note that $u \prec_{\pi_i} v$ is also defined if u and/or v are not leaves). Then $\pi'_1, \pi'_2, \dots, \pi'_\ell$ witnesses the planarity of \mathcal{T} .

“ \Leftarrow ”: Assume that \mathcal{T} is planar. Thus, there exist permutations $\pi_1, \pi_2, \dots, \pi_\ell$ of the leaves of the trees T_1, T_2, \dots, T_ℓ with the required properties. For each π_i do the following to obtain the permutation π'_i of the leaves in $\mathcal{G}|_{t_i}$. For each node u in π_i , if the parent of u is a leaf v in the tree $\mathcal{G}|_{t_i}$, replace u with v ; otherwise remove u from π_i . The obtained combinatorial layout $\mathcal{L} = (\pi'_1, \pi'_2, \dots, \pi'_\ell)$ of \mathcal{G} is planar: Suppose \mathcal{L} is not planar, hence two temporal edges e, f cross in \mathcal{L} . As \mathcal{G} is hierarchy-compliant, there must exist two leaf edges e', f' that cross in \mathcal{L} . Let $e' = (u, v)$ and $f' = (w, x)$. We have that the edges (u', v') and (w', x') must have already crossed in \mathcal{T} by construction, a contradiction.

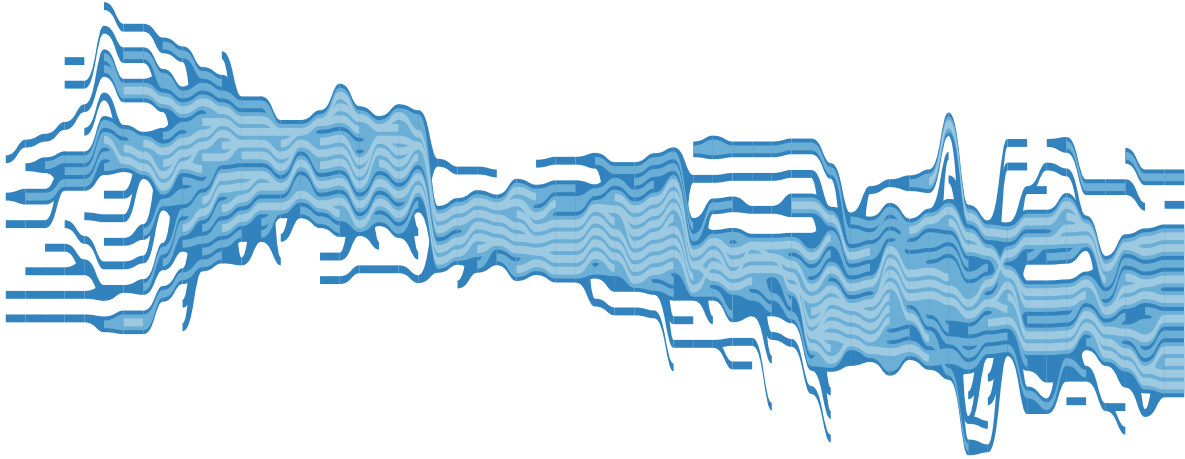
It follows, that we can use polynomial-time \mathcal{T} -level planarity testing algorithms [ALB⁺15, JLM98] to test whether a hierarchy-compliant temporal tree is planar. \square

3 Case Study

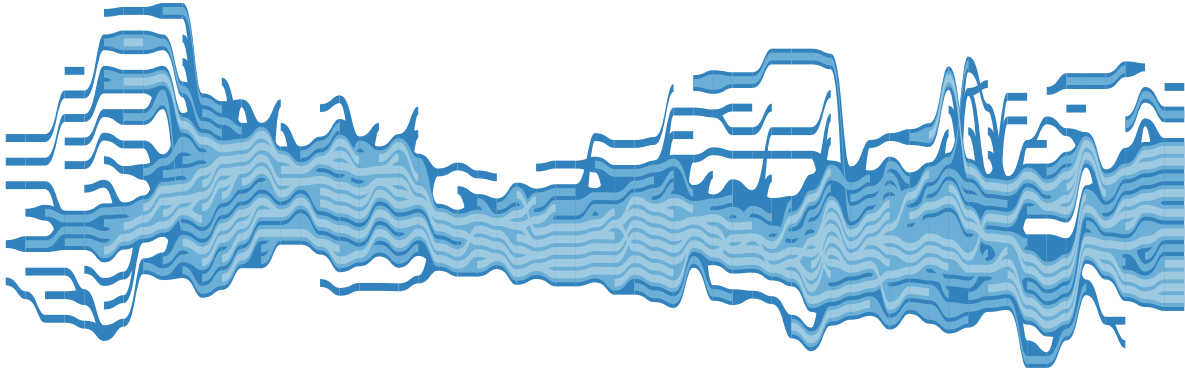
Here, we want to give more examples of temporal treemaps that result from the layouts computed by the different algorithms – our heuristics, integer linear programs, and the algorithm of Köpp and Weinkauff [KW19]. For each example we give in the captions information about the instance, and about the algorithm. In particular we give the number of nodes, leaves, and depth of the input temporal tree. For the algorithm we give its runtime, the resulting number of crossings and number of leaf edge crossings. The layouts produced by our heuristic algorithms, BARY, MEDI, BARY-LE, and MEDI-LE, generally look very similar, which is why we only show the layout computed by MEDI. The same goes for ILP and ILP-LE; we only show layouts computed by ILP.

3.1 Viscous Fingers

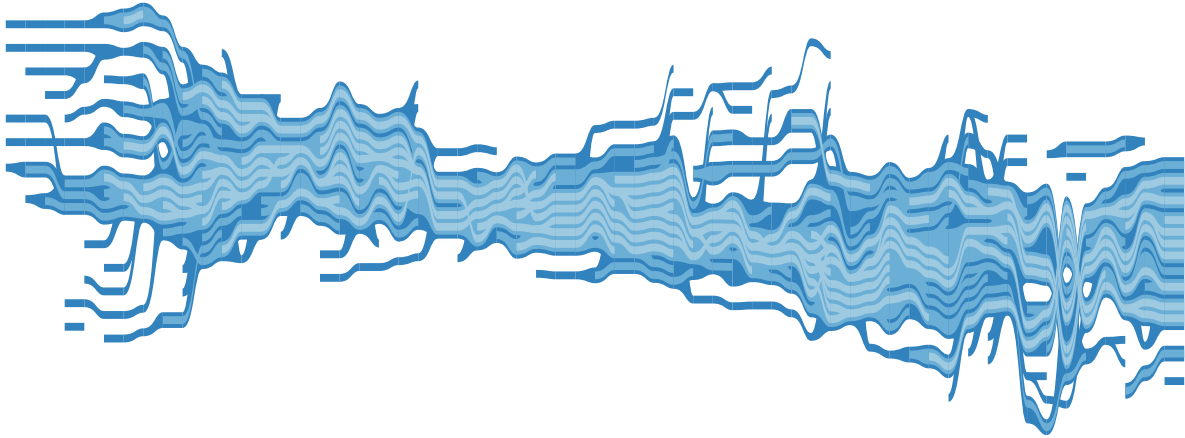
Fig. 2 and Fig. 3 show temporal treemap layouts computed by the different algorithms for the Viscous Fingers data set which has already been discussed in the paper. The difference between the two figures is that in Fig. 2 we have removed the node weights of the temporal tree for the edge crossings to be more visible. Note that the layouts between Fig. 2 and Fig. 3 can differ as all considered algorithms are not deterministic and the visualizations show different runs of the algorithms.



(a) ILP: 10 crossings, 7 leaf crossings, 332ms



(b) MEDI: 83 crossings, 58 leaf crossings, 20ms



(c) KW: 215 crossings, 109 leaf crossings, 59000ms

Figure 2: Combinatorial layouts computed for the Viscous Fingers dataset by different algorithms shown as temporal treemaps. Captions report number of crossings, leaf crossings, and runtime. The input temporal tree has 939 nodes, 670 leaves, and depth 3. The weights of the temporal tree nodes have been removed.

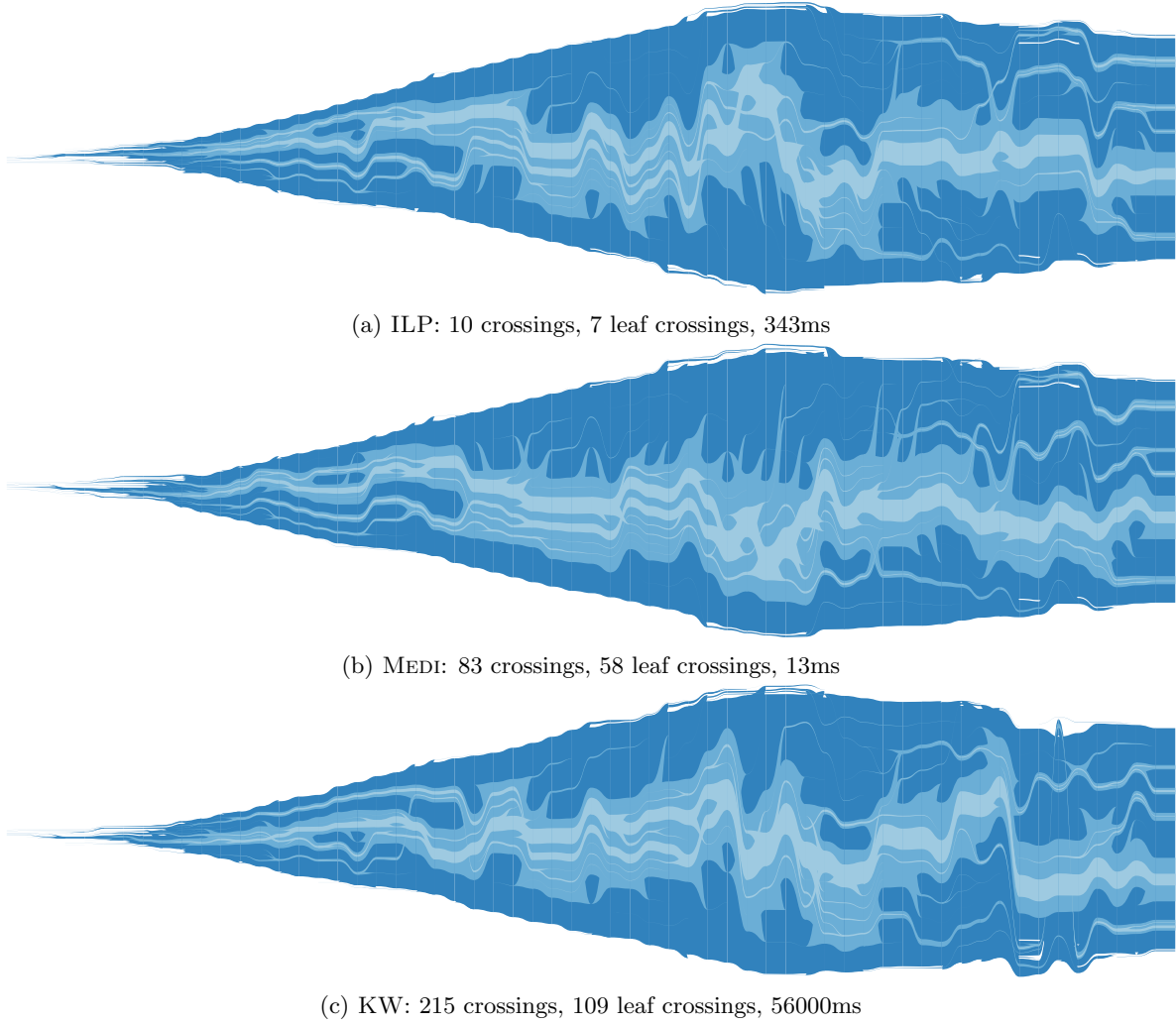


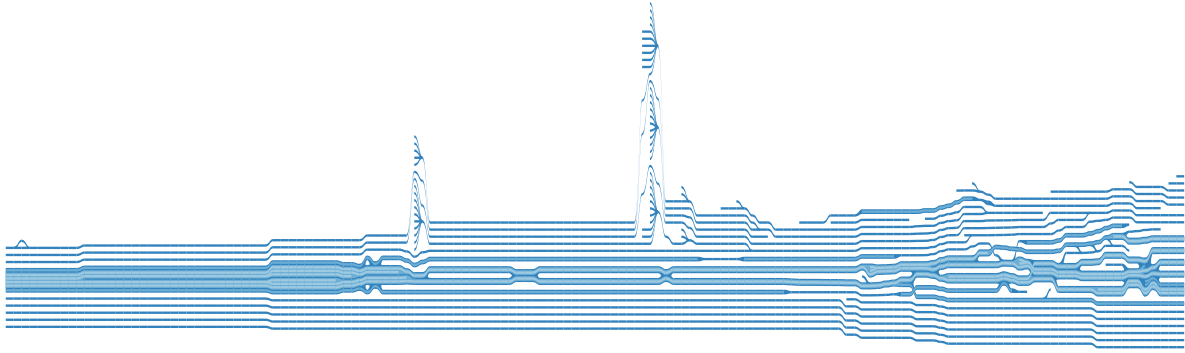
Figure 3: Combinatorial layouts computed for the Viscous Fingers dataset by different algorithms shown as temporal treemaps. Captions report number of crossings, leaf crossings, and runtime. The input temporal tree has 939 nodes, 670 leaves, and depth 3.

3.2 Cylinder

Next is the Cylinder dataset describing vortex activity in the wake of a square cylinder by means of the Okubo-Weiss criterion [CBI⁺05, vFWTS08]. Computing a combinatorial layout is only possible for our heuristic algorithms such as MEDI (this results in 8500 crossings in time $372ms$), for KW and ILP the execution times out. However, we cannot visualize this combinatorial layout as standard browsers run into a memory limit for this visualization.

In our experiments we have split up the Cylinder dataset into multiple smaller data sets by taking specific ranges of time steps. We show visualizations for two such smaller datasets below. Again, we remove the node weights for the visualizations to be more readable and to better recognize crossings.

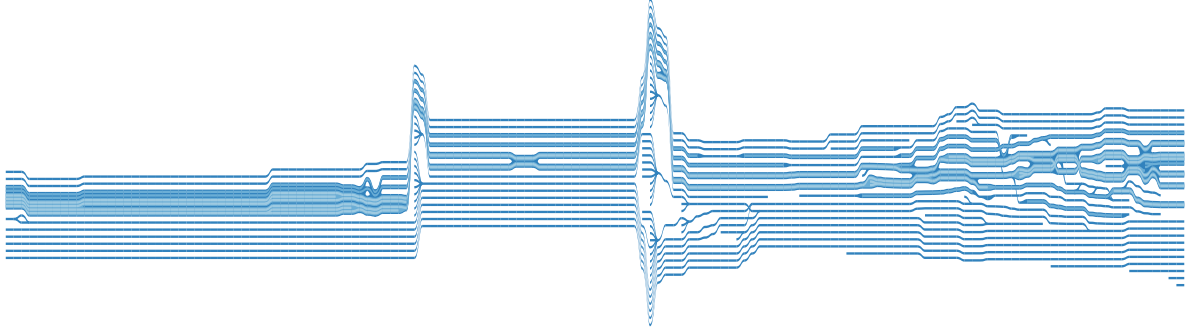
Range 20-169. We first show layouts for the range 20-169 in Fig. 4.



(a) ILP: 14 crossings, 5 leaf crossings, 8473ms



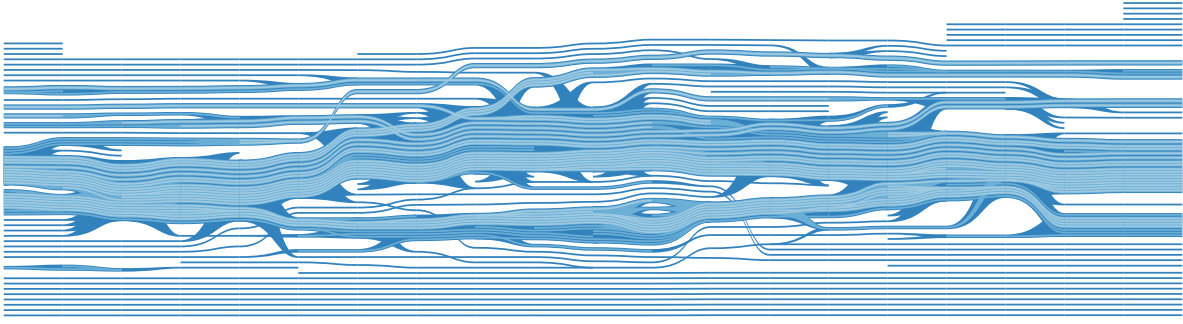
(b) MEDI: 20 crossings, 7 leaf crossings, 56ms



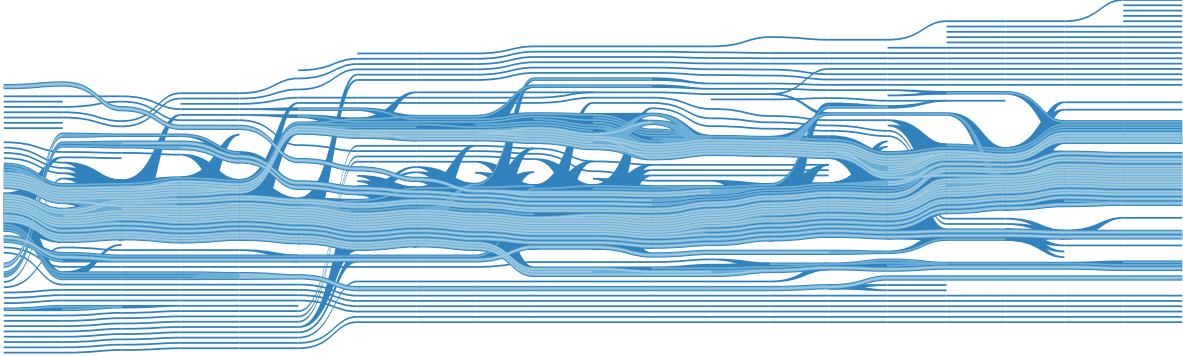
(c) KW: 45 crossings, 17 leaf crossings, 3094ms

Figure 4: Combinatorial layouts computed for the Cylinder dataset for range 20-169 by different algorithms shown as temporal treemaps. Captions report number of crossings, leaf crossings, and runtime. The input temporal tree has 3248 nodes, 2440 leaves, and depth 3. The weights of the temporal tree nodes have been removed.

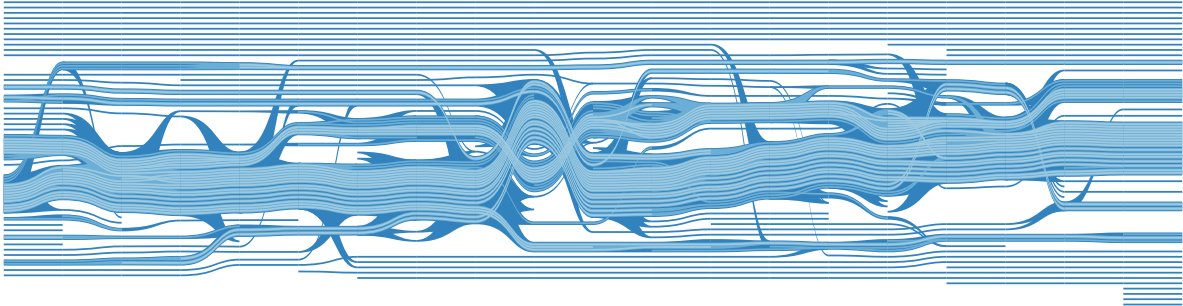
Range 400-419. Fig. 4 shows the range 400-419.



(a) ILP: 209 crossings, 113 leaf crossings, 188286ms



(b) MEDI: 1048 crossings, 593 leaf crossings, 25ms



(c) KW: 2470 crossings, 1385 leaf crossings, 152784ms

Figure 5: Combinatorial layouts computed for the Cylinder dataset for range 400-419 by different algorithms shown as temporal treemaps. Captions report number of crossings, leaf crossings, and runtime. The input temporal tree has 1489 nodes, 1142 leaves, and depth 3. The weights of the temporal tree nodes have been removed.

References

- [ALB⁺15] Patrizio Angelini, Giordano Da Lozzo, Giuseppe Di Battista, Fabrizio Frati, and Vincenzo Roselli. The importance of being proper: (in clustered-level planarity and T-level planarity). *Theor. Comput. Sci.*, 571:1–9, 2015.
- [CBI⁺05] Simone Camarri, M Buffoni, A Iollo, MARIA VITTORIA Salvetti, et al. Simulation of the three-dimensional flow around a square cylinder between parallel walls at moderate reynolds numbers. In *XVII Congresso AIMETA di Meccanica Teorica e Applicata, Volume II*, volume 1, pages 23–34. Firenze University Press, 2005.
- [JLM98] Michael Jünger, Sebastian Leipert, and Petra Mutzel. Level planarity testing in linear time. In Sue Whitesides, editor, *Proc. Graph Drawing and Network Visualization (GD’98)*, volume 1547 of *Lecture Notes in Computer Science*, pages 224–237. Springer, 1998.

- [KW19] Wiebke Köpp and Tino Weinkauff. Temporal treemaps: Static visualization of evolving trees. *IEEE Trans. Vis. Comput. Graph.*, 25(1):534–543, 2019.
- [vFWTS08] Wolfram von Funck, Tino Weinkauff, Holger Theisel, and Hans-Peter Seidel. Smoke surfaces: An interactive flow visualization technique inspired by real-world flow experiments. *IEEE Trans. Vis. Comput. Graph.*, 14(6):1396–1403, 2008.