A Cycle Basis Algorithm

Here we present pseudocode for our tight cycle basis algorithm described in Section 4.2 of the main paper. The input is a topological block T of some hypergraph H and the output is a basis $\mathscr{C}$ for the cycle space of T containing only tight cycles. Let $n = |V(T)|$ and let $m = |E(T)|$. The complexity of our tight cycle basis algorithm is $O(B_1(T)(n+m))$ where $B_1(T)$ is the first Betti number. The Euler characteristic gives us $B_1(T) = 1+m-n$, so the complexity is in fact $O((m-n)(n+m)) = O(m^2-n^2)$.

Algorithm 1: BFS Tight Cycle Basis

Input: Topological block T
Output: Tight cycle basis $\mathscr{C}$ of T

function BfsBasis(T)
  $Q \leftarrow$ initialize queue ; // outer search queue
  $S \leftarrow$ empty subgraph ; // visited nodes and traversed edges
  choose node $x_0 \in V(T)$;
  enqueue $x_0 \to Q$; insert $x_0 \to V(S)$;
  while $|Q| > 0$ **do** // outer BFS loop
    $x \leftarrow Q$.front; Q.pop;
    for *each edge* $(x,y) \in E(T)$, $(x,y) \notin E(S)$ **do**
      if $y \in V(S)$ **then** // cycle detected
        $C =$ TightCycle(S,x,y) ; // call inner BFS
        add C to homology basis $\mathscr{C}$;
      else
        enqueue $y \to Q$; insert $y \to V(S)$;
      insert $(x,y) \to E(S)$;
  return $\mathscr{C}$;
function TightCycle(S,x,y)
  $Q \leftarrow$ initialize queue ; // inner search queue
  $U \leftarrow$ empty subgraph ; // visited nodes and traversed edges
  enqueue $x \to Q$; insert $x \to V(S)$;
  x.parent $\leftarrow$ null;
  while $|Q| > 0$ **do** // inner BFS loop
    $a \leftarrow Q$.front; Q.pop;
    for *each edge* $(a,b) \in E(S)$, $(a,b) \notin E(U)$ **do**
      if $b = y$ **then** // tight cycle found
        $C \leftarrow$ new cycle; insert $(a,y) \to C$;
        $p \leftarrow a$.parent;
        while $p \neq$*null* **do**
          insert $(p,a) \to C$;
          $a \leftarrow p$; $p \leftarrow p$.parent;
        return C;
      else
        enqueue $b \to Q$; insert $b \to V(U)$;
        b.parent $\leftarrow a$;
      insert $(a,b) \to E(U)$;

B Proof of Theorem 3

Here we prove Theorem 3 from the main paper which we restate below.

Theorem 4. *(Theorem 3 from the main paper) A cycle in $A_{(\mathscr{C}4)}$ defined by the sequence $F = \langle C_1, C_2, \ldots, C_k \rangle \subseteq \mathscr{C}_4$ for some tight cycle basis $\mathscr{C}$ of T contains a common primal or dual node $x \in V(G_T)$ within each of the basis cycles $C_i \in F$ iff F corresponds to a forbidden sub-hypergraph in* H.

Proof. $\Leftarrow$ Since the basis cycles in $\mathscr{C}$ are linearly independent, a subset of minimal basis cycles, all having length 4, can have at most 3 common elements. Let S denote the set of common elements for each $C_i \in F$ and assume that $|S| \geq 1$. We show that for any possible combination of elements in S, F contains one of the forbidden sub-hypergraphs.

In the case that $|S| = 1$, the sole element in S is either a primal or dual node of G_T. Without loss of generality, suppose that the sole element

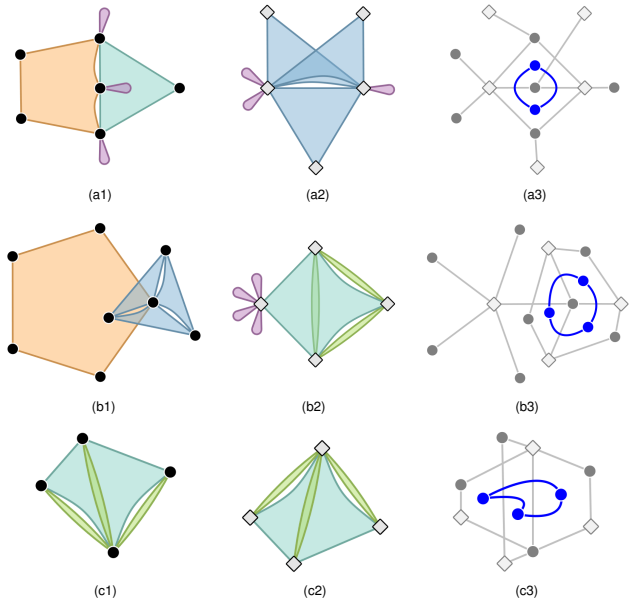
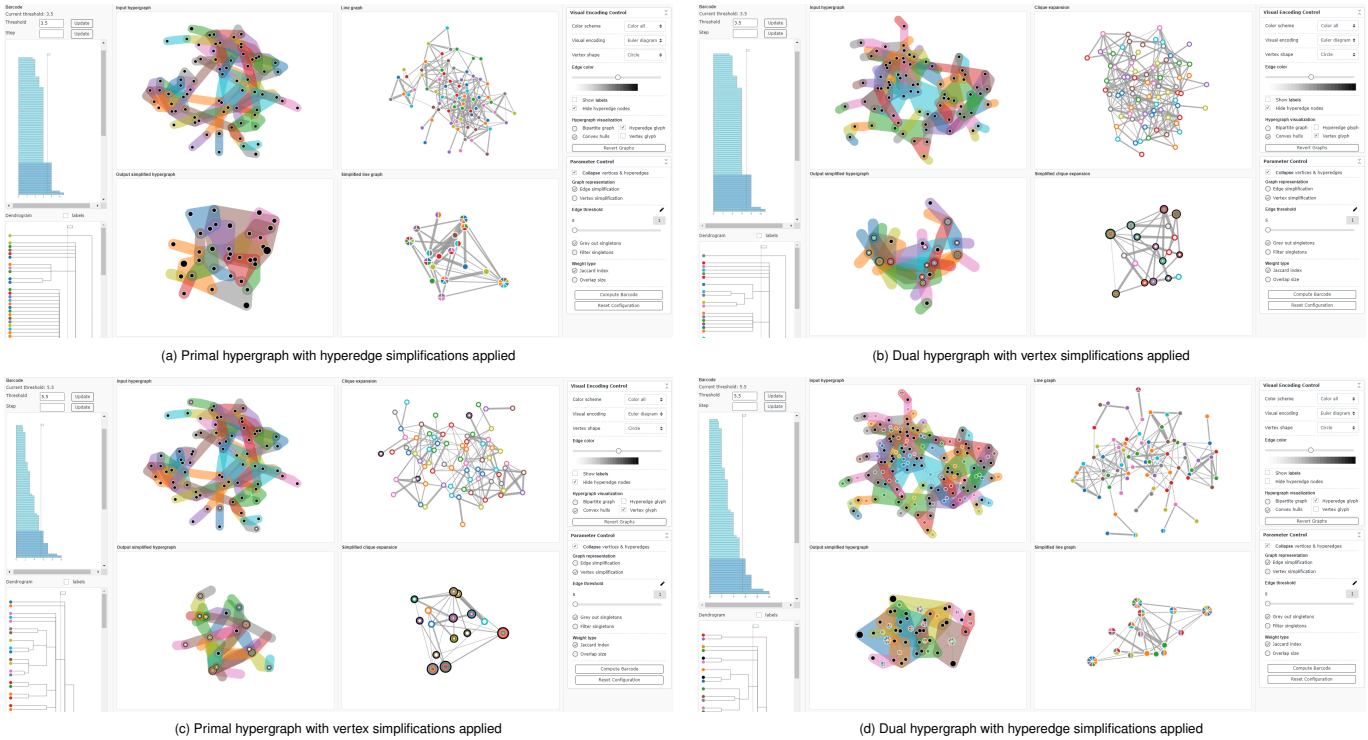


Fig. 10: The forbidden sub-hypergraphs of polygon hypergraph drawings: (a1) 3-adjacent hyperedge bundle of 2 hyperedges, (a2) 2-adjacent hyperedge bundle of 3 hyperedges, (b1) strangled vertex cycle variant, (b2) strangled hyperedge cycle variant, (c1,c2) strangled vertex and hyperedge star variant. Notice that (a2) is the dual of (a1), (b2) is the dual of (b1), and (c2) is the dual of (c1). The cycle adjacency graph for each primal-dual pair is drawn in blue over the corresponding bipartite graph representation in (a3), (b3), and (c3).

$v_0 \in S$ is a primal node corresponding to a vertex in the primal hypergraph. For F to form a cycle in $A(\mathscr{C}_4)$, it must be that each consecutive pair of basis cycles $C_i, C_{i+1} \in F$ share an edge (v_0, e_i) in G_T where each $e_i \in V(G_T)$ is a dual node. Then each cycle $C_i \in F$ must contain elements $\langle e_{i-1}, v_0, e_i, v_i \rangle$ where $v_i \in V(G_T)$ is a primal node not equal to v_0. It follows that the sequence $\langle v_1, e_1, v_2, e_2, \ldots, v_k, e_k, v_1 \rangle$ defines a cycle in T. Since this sequence only contains hypergraph elements that are incident and adjacent to the primal vertex v_0, v_0 matches the definition of the cycle variant of a strangled vertex (Figure 10 (b1,b3)). Similarly, if the sole element in $e_0 \in S$ is a dual node of G_T, corresponding to a hyperedge in the primal hypergraph, we can show that e_0 matches the definition of the cycle variant of a strangled hyperedge (Figure 10 (b2,b3)).

Now consider the case where S contains exactly one primal node $v_0 \in V(G_T)$ and one dual node $e_0 \in V(G_T)$. For F to form a cycle in $A(\mathscr{C}_4)$, it must be that $|F| = k \geq 3$ since two cycles $C_i, C_j \in F$ sharing the edge (v_0, e_0) in G_T corresponds to a single edge in $A(\mathscr{C}_4)$ which is not a cycle. Then each basis cycle $C_i \in F$ must also contain primal and dual nodes $v_i, e_i \in E(G_T)$ where v_i and e_i are incident to each other, v_i is incident to e_0, and e_i is incident to v_0. It follows that the set $\{e_0, v_1, e_1, \ldots, v_k, e_k\}$ induces a star sub-hypergraph in T where e_0 is the central element and each sequence $\langle e_0, v_i, e_i \rangle$ for $i \geq 1 \leq k$, $k \geq 3$, defines a point of the star. Similarly, the set $\{v_0, e_1, v_1, \ldots, e_k, v_k\}$ induces a star sub-hypergraph centered on v_0 where each sequence $\langle v_0, e_i, v_i \rangle$ defines a point of the star. Since both of these sets only contain hypergraph elements incident and adjacent to the primal vertex v_0 and primal hyperedge e_0 respectively, v_0 matches the definition of the star variant of a strangled vertex, and e_0 matches the definition of the star variant of a strangled hyperedge (Figure 10, (c1,c2,c3)).

Finally, consider the case where $|S| \geq 2$ and S contains either a pair of primal nodes or a pair of dual nodes in G_T. Without loss of generality, suppose that S contains a pair of primal nodes $v_0, v_1 \in$ V. Then for F to form a cycle in $A(\mathscr{C}_4)$, each consecutive pair of basis cycles $C_i, C_{i+1} \in F$ must share a dual node $e_i \in V(G_T)$ as well as edges $(v_0, e_i), (v_1, e_i)$. This means that each pair of consecutive basis cycles is connected by two edges in $A(\mathscr{C}_4)$, and each sub-sequence $\langle C_i, C_{i+1} \rangle$ is also a cycle



(a) Primal hypergraph with hyperedge simplifications applied

(b) Dual hypergraph with vertex simplifications applied

(c) Primal hypergraph with vertex simplifications applied

(d) Dual hypergraph with hyperedge simplifications applied

Fig. 11: Screen captures from the interactive tool presented in Zhou *et al.* [38] with both the primal and dual hypergraphs of the museum interactions dataset from [15]. Inside each screen capture, the input hypergraph and graph representation are shown on the top row of visualizations, the persistence barcode with simplification threshold applied on the left, and the simplified hypergraph and graph representation on the bottom row of visualizations.

in $A(\mathscr{C}_4)$. Considering only one of these sub-sequences, we have $C_i = \langle v_0, e_i, v_1, e_\alpha \rangle$ and $C_{i+1} = \langle v_0, e_i, v_1, e_\beta \rangle$ where $e_\alpha, e_\beta \in \mathsf{E}$ are distinct. Then e_i, e_α, e_β are all adjacent hyperedges with two common elements v_0, v_1, matching the definition for a 2-adjacent bundle of 3 hyperedges. We can further show that the original sequence F contains elements matching the definition of a 2-adjacent bundle of $k+1$ hyperedges. If S instead contains a pair of hyperedges $e_0, e_1 \in \mathsf{E}$, we can similarly show that each sub-sequence $\langle C_i, C_{i+1} \rangle$ contains elements matching the definition of a 3-adjacent bundle of 2 hyperedges and F contains elements matching the definition of a $(k+1)$-adjacent bundle of 2 hyperedges.

$\Rightarrow$ From Figure 10 (a3,b3,c3), it is straightforward to verify that we can construct a minimum cycle basis for each of the forbidden sub-hypergraphs that satisfies the conditions of Theorem 4. $\square$

C Comparison to Zhou *et al.* [38]

Here we provide a comparison of our structure-aware hypergraph simplification framework to the persistent homology guided simplification framework of Zhou *et al.* [38].

Zhou *et al.* use a line graph or clique expansion graph as the computational representations for the input hypergraph H, applying edge collapse operations on the chosen graph representation to induce hyperedge collapse or vertex collapse operations in H respectively. The line graph representation has a node set corresponding to the hyperedges of H with edges representing non-zero intersection between the vertex sets of the corresponding pair of hyperedges. These edges are weighted using the reciprocal of either the size of the intersection between the pair of hyperedges, or the Jaccard index between their vertex sets. The clique expansion graph representation has a node set corresponding to the vertices of H with edges representing containment of the pair of corresponding vertices in at least one common hyperedge. Again, the edges are weighted using the reciprocal of either the number of common hyperedges or the Jaccard index of the containing hyperedge sets for the corresponding pair of vertices. They then apply persistent homology to a metric space of the chosen graph representation G to generate a barcode where each bar represents an edge in a minimum spanning tree (MST) of G. The length of the bars corresponds to the edge weights. They collapse edges in the MST G if the length of the corresponding bar in the barcode is less than a user controllable threshold, thereby inducing hyperedge or vertex simplifications in H.

By using separate graph representations, their approach is not able to generate simplified results that include both hyperedge and vertex simplifications simultaneously. Furthermore, the line graph and clique expansion graph representations are different depending on whether the primal hypergraph H or dual hypergraph H^* is used. In fact, the line graph of H is equivalent to the clique expansion graph of H^* and vice versa. By using the bipartite graph as our computational representation of H, which treats vertices and hyperedges in the same way, we naturally include hyperedge and vertex simplifications in a single unified framework. As discussed in our main paper, the bipartite graph representation $G(\mathsf{H})$ is equivalent to $G(\mathsf{H}^*)$ and can be regarded as the same graph. Thus, our approach treats the primal and dual hypergraphs with equal importance while Zhou *et al.* [38] may introduce inconsistency depending on the input and which graph representation is chosen.

Figure 11 shows the museum interactions hypergraph, from Section 7 of the main paper obtained from the dataset of Isella *et al.* [15], loaded into the interactive tool presented by Zhou *et al.* [38]. We accessed the tool from their GitHub repository at `https://github.com/tdavislab/Hypergraph-Vis/tree/master`. We tested their simplification method using both the hyperedge and vertex simplification options applied to both the primal hypergraph H and dual hypergraph H^* of the original dataset, visualizing the results in the default Euler diagram style. We adjusted the simplification threshold in all four cases to find the smallest value that would produce a Zykov planar hypergraph, i.e., a hypergraph whose bipartite representation is a planar graph. We note that there are no additional layout options for the Euler diagram visualizations other than the default output of their system. This makes it difficult to observe any structures in the data, like

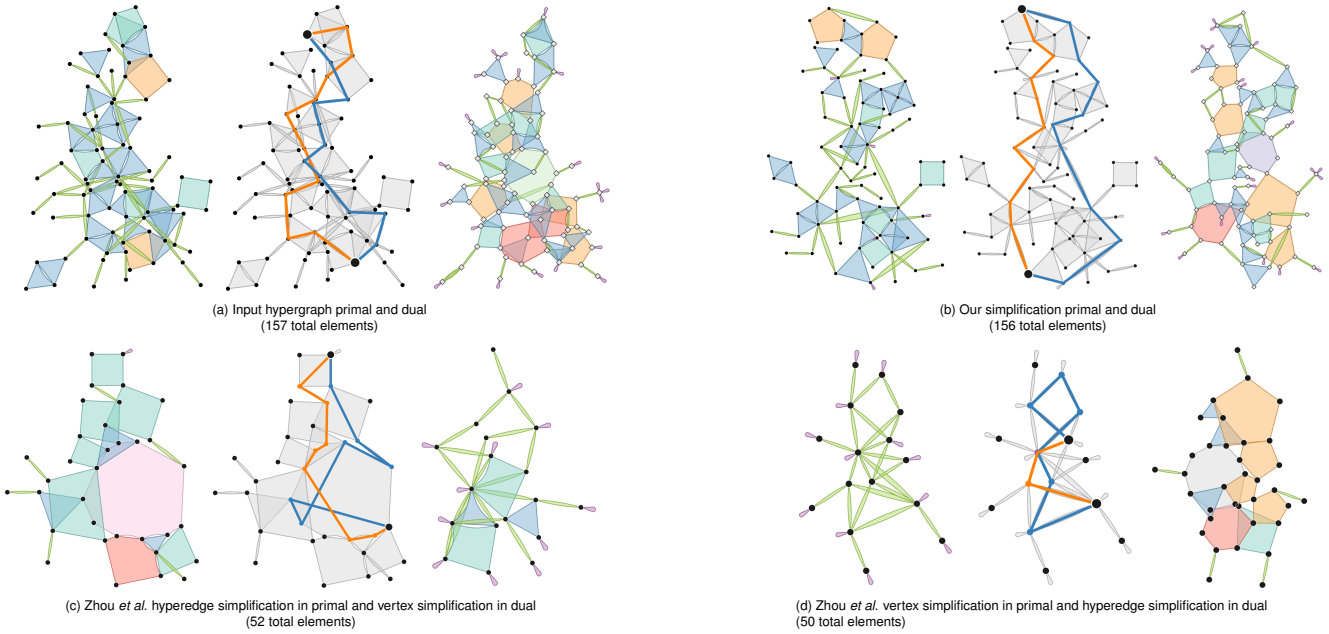


Fig. 12: A comparison of simplification results from Zhou *et al.* [38] and ours for the museum interactions dataset from [15]. Each sub-figure includes the primal and dual hypergraphs with polygons colored according to hyperedge cardinality on the left and right. In the middle, the same pair of hypergraph paths are highlighted in the primal hypergraph of each sub-figure indicated by orange and blue lines. These paths are disjoint in the original hypergraph (a) except for the start and end points.

cycles, bridges, and branches unless the layout is manually adjusted. However, it is convenient that the input hypergraph and simplified result are displayed next to each other.

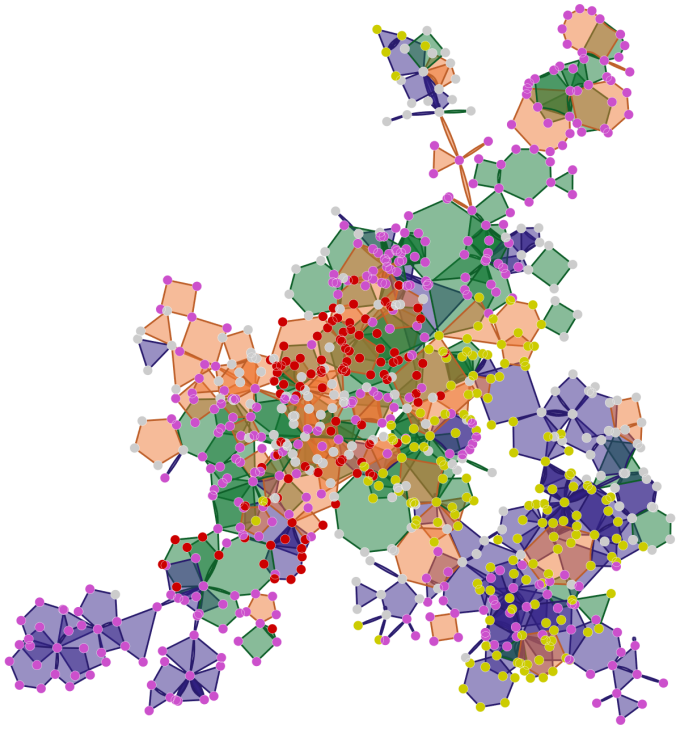
Since the line graph representation of H is equivalent to the clique expansion graph of H^* and vice versa, the vertex simplified result in Figure 11 (b) is the dual hypergraph of the hyperedge simplified result in (a), and the hyperedge simplified result in (d) is the dual hypergraph of the vertex simplified result in (c). This is also indicated by the fact that the barcode of (a) matches the barcode of (b) and the barcode of (c) matches the barcode of (d). However, since the primal and dual simplifications can only be performed separately, it is not possible to coordinate the layouts of the results, which could lead to different interpretations of the data and its structural features depending on which is used. More specifically, the upper left sub-windows in Figure 11 (a) and (b) provide seemingly unrelated input visualizations, even though one is based on the primal hypergraph and the other on the dual hypergraph of the same data. Furthermore, between Figure 11 (a) and (c), even though the visualization of the input hypergraph (upper left sub-windows) is the same, the simplified results (the sub-windows below the input visualizations) are different and seemingly unrelated. This is due to the different types of simplification operations that are used: (a) hyperedge simplification based on the clique expansion graph representation, and (c) vertex simplification based on the line graph representation. In contrast, we use the bipartite graph representation to handle the primal hypergraph and its dual in a consistent fashion.

We also compare their simplifications to ours using the polygon visualization metaphor. We exported the simplified hypergraphs from Zhou et al.'s tool and imported them into our polygon visualization tool which is available at `https://github.com/tdavislab/Hypergraph-Vis/tree/master`. Figure 12 compares these visualizations of the original museum interactions hypergraph (a), our simplified result from the main paper Figure 9 (b), the hyperedge simplification of the primal hypergraph from Zhou *et al's.* tool [38] paired with their vertex simplification of the dual hypergraph (c), and the vertex simplification of the primal hypergraph from Zhou *et al's.* tool (d) paired with their hyperedge simplification of the dual hypergraph, all using the polygon visualization metaphor. The dual hypergraph visualizations in (a) and (b) are coordinated with the primal visualizations such that corresponding primal and dual elements appear in approximately the same locations. Since they are generated separately, the dual visualizations in (c) and (d) are not coordinated with the primal visualizations, resulting in visualizations that may look unrelated. We observe that both the hyperedge and vertex simplification methods of Zhou *et al.* are effective in reducing the size of the data while keeping some of the original features, but much of the data is simplified away to reach a result that is Zykov planar, keeping only 50–52 elements compared to the original 157. Our result, on the other hand, keeps 156 elements, retaining most of the original information and also preserving the path structures indicated by the orange and blue highlights in the middle visualization of each sub-figure. These highlights represent a pair of paths that are disjoint in the original hypergraph Figure 12 (a) except at the starting and ending vertices. In (b), we can clearly see that these paths are still distinct. In (c) however, the paths intersect through a common super-hyperedge, and in (d), they intersect through a common super-vertex. Additionally, the hyperedge and vertex simplification methods alter the paths in different ways as we can see with the orange path in (d) which is significantly shorter than the orange path in (c). For a visualization application that requires identifying disjoint paths, such as identifying transmission vectors of an infectious disease, we argue that our simplification provides a benefit over Zhou *et al's.* .

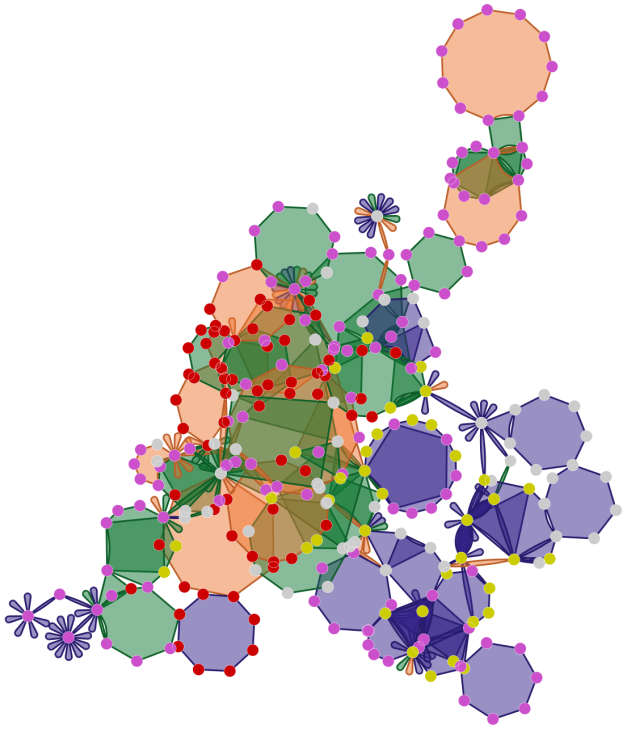
A more thorough comparison of Zhou *et al's.* [38] with ours is challenging because of the difference in simplification goals. We aim to reduce clutter by reducing unavoidable overlaps while preserving structural information. Zhou *et al.* aim to reduce visual clutter by reaching a compact representation that can be used for analysis. Both approaches have applications for which they may be more appropriate. Additionally, we cannot directly compare results based on the number of simplifications applied or the number of elements in the simplified results because the frameworks have different termination criteria. Ours is based on the removal of unavoidable overlaps and theirs is based on a persistence threshold value, i.e., if a set of features have the same barcode length, they are either all simplified or all not simplified.

D Enlarged Figures

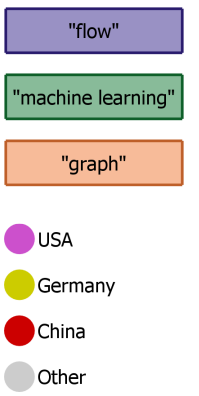
On the following pages, we provide enlarged versions of Figure 8 and Figure 9 from the main paper.

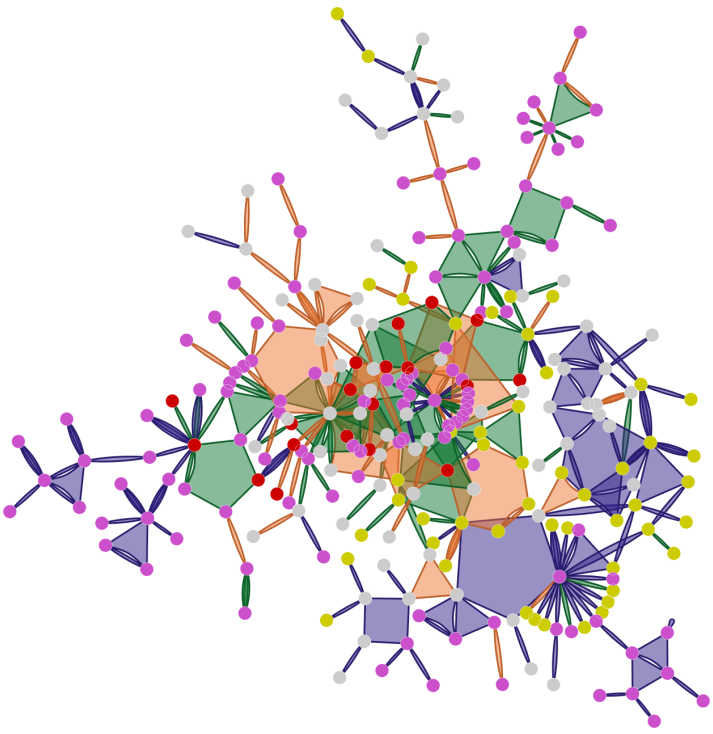


(a) Input hypergraph



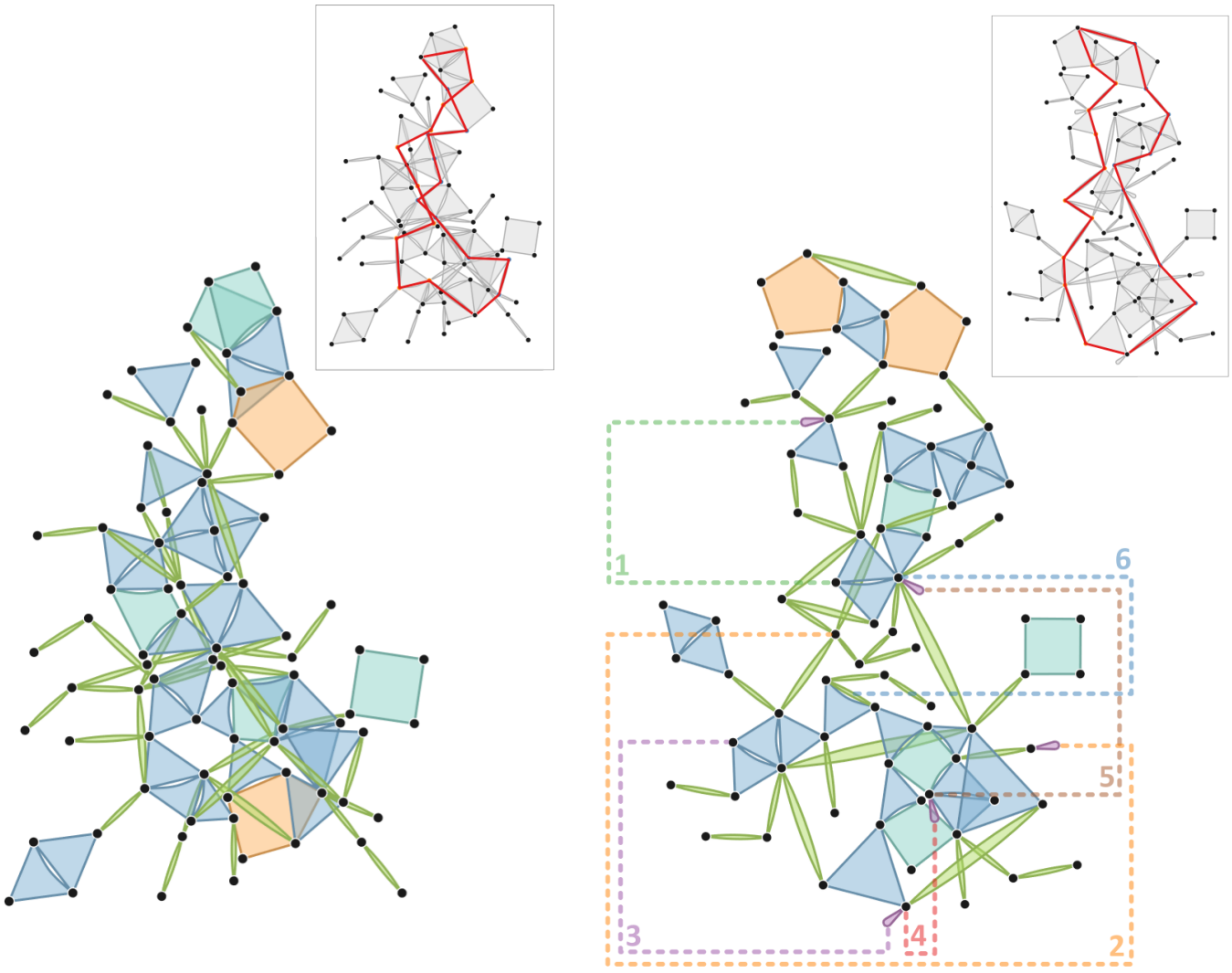
(b) Oliver *et al.* simplified scale





(c) Our simplified scale

Fig. 13: Enlarged versions of the visualizations in Figure 8 from the main paper. A paper-author hypergraph network with 786 vertices and 318 hyperedges, (a) is simplified using the priority guided approach of Oliver *et al.* [25] (b), and our topological decomposition guided approach (c). Notice that some of the branches in (b) have been reduced and others have been eliminated entirely. Our hypergraph decomposition extracts structures ahead of time, allowing us to preserve the skeleton of each branch. In each visualization, the hyperedges are colored according to the keywords of their papers: blue for the keyword “flow”, green for the keywords “machine learning”, and orange for the keyword “graph”. Additionally, the vertices are colored according to the geographic location of the affiliated research institution for each author: magenta for institutions based in the United States, yellow for institutions based in Germany, red for institutions based in China, and gray for institutions based elsewhere.



(a) Input hypergraph

(b) Simplified hypergraph with cycle edge cuts

Fig. 14: Enlarged versions of the visualizations in Figure 9 from the main paper. A hypergraph representing face-to-face interactions between museum visitors [15] is visualized before, (a), and after simplifying the topological blocks, (b). Notice the numerous triangles and digons in (a) that overlap despite being non-adjacent. On the right, six cycles have been cut to make the hypergraph planar. The deleted incidence relationships between vertices and hyperedges are represented by dashed annotation lines along the boundary of the visualization. In the upper right corners, we highlight the same two paths for each layout from a vertex near the top to a vertex near the bottom.