Trimming the Hairball: Edge Cutting Strategies for Making Dense Graphs Usable

Darren Edge
Microsoft AI & Research
Microsoft
Cambridge, UK
daedge@microsoft.com

Jonathan Larson
Microsoft AI & Research
Microsoft
Silverdale, WA, USA
jolarso@microsoft.com

Markus Mobius
Microsoft AI & Research
Microsoft
Cambridge, MA, USA
mobius@microsoft.com

Christopher White
Microsoft AI & Research
Microsoft
Redmond, WA, USA
chwh@microsoft.com

***Abstract*—The application of modern NLP and ML techniques to large-scale datasets can generate implicit graphs that are so densely connected as to be unusable when rendered as node-link diagrams. We present a two-stage approach to extracting usable, map-like layouts from large, dense input graphs. This approach uses edge-cutting strategies based on node and edge metrics to reduce a graph to a skeletal structure showing only essential relationships, before filling in the resulting communities to create dense but usable layouts. Through a case study on a 145k-document adversarial health communication dataset, we show that each edge-cutting strategy has advantages and disadvantages, and that the appropriate choice of strategy depends on the data, user, and task.**

Keywords—implicit graphs, graph representations, node-link diagrams, deterministic edge filtering, visual analytics

I. Introduction

Node-link diagrams have remained the archetypal representation of entity relationships since the thirteenth century. Their combination of familiarity and interpretability make them accessible to a general audience and they are often the default choice for rendering graph/network/topology data. At the same time, conventional wisdom in information visualization is that node-link diagrams do not scale to large or densely-connected graphs because the resulting link crossings, link distances, and node overlaps [10][36] make it difficult to complete typical graph tasks: counting all nodes and links, counting the links of a node, following links to their targets; and finding paths between nodes of interest [10][27].

While much research has focused on the design of alternative graph representations (and renderings) that either reduce or eliminate link crossing, we argue for an emerging class of graph use cases in which individual links, link crossings, and link-based tasks are of reduced importance. This is the often the case when links are not given by a "real" network (e.g., an explicit transportation, communication, citation, or social network), but derived through data mining, machine learning, or other statistical approaches to inferring relations between things (i.e., an implicit or hidden network).

A key domain for derived graphs is text analytics, in which linguistic units (e.g., words, phrases) are extracted from text, and edges are used to encode the frequency with which pairs of units co-occur across a corpus. The use of graphs to automatically summarize machine-readable text has been practiced since the bulletin boards of the pre-web era [6], forms a key part of visual analytics tools (e.g., [43]), and is highly applicable to adversarial use cases such as mining covert networks from text [8]. Such mined relations may not always correspond to meaningful relationships, however.

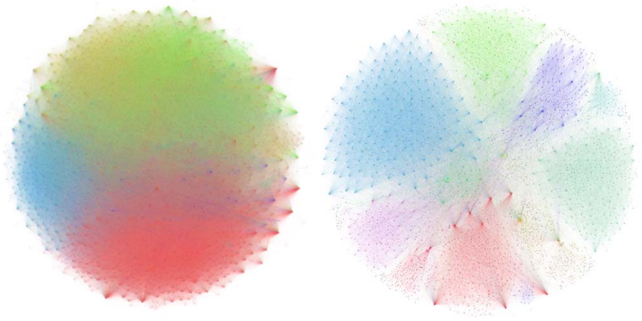

Fig. 1. Left: "Hairball" entity-relationship graph with 4k nodes densely interconnected by 200k links ($Q = 0.26$). Right: results from cutting edges to a density of 5.0 edges per node in order of increasing mutual information, then reintroducing all within-community edges ($Q = 0.77$).

When edges represent what might be, rather than what is, it becomes less important to faithfully render all such edges as links, and more important to reveal groups of related nodes as broad-brush summaries of underlying phenomena (e.g., topics of discussion on web forums). Such derived graphs act more like maps for navigating the territory of the given data, in ways that challenge conventional notions of graph tasks. Recent empirical studies have also shown that the negative impact of link crossing diminishes with increasing graph size [24], that graphs rendered as map-like node-link-group diagrams [12] increase memorability [38] and performance on group-level tasks with no negative impact on node-level tasks [39], and that adding map-like substrates and visual landmarks to node-link diagrams supports orientation and revisitation [13]. The implication is that not only can node-link diagrams be used as abstract maps for navigating linked documents or other data objects, but that the more they look like actual maps, the more useful and usable they will be.

The challenge addressed in this paper is that many graphs, especially large derived/implicit graphs, rarely have map-like qualities when rendered using standard force-directed layout algorithms. A more typical visual impression is one of a giant "hairball", with significant node occlusion and link crossings that can almost completely fill the inter-node space (Fig. 1). Such structures arise from a combination of factors, including high-degree nodes that reduce the diameter of the graph, low-weight links that increase the density of the graph, the scale of the raw data represented by the graph, and the errorful data processing mechanisms that generate the graph structure. By "trimming" low-signal edges from such graphs, we can help mitigate the effects of a noisy graph generation process in ways that scale to real-world datasets. The goal is for nodes to achieve both clear separation and grouping (e.g., community clustering) given an appropriate layout algorithm.

We achieve this goal through a two-stage approach to generating usable, map-like layouts from even the densest of input graphs. The first stage is to use a deterministic edge-cutting strategy to reduce the input graph to a skeletal structure of essential relationships within and between communities. The second stage is to reinstate edges from the input graph that represent intra-community edges in the skeletal graph. We illustrate this approach through a case study analyzing a large-scale, naturalistic dataset based on named entity co-occurrence in news articles that may be related to the vaccination debate. This is a well-known target for communication that is both automated and adversarial, and sometimes referred to as "weaponized health communication" [5]. In this use case, analysts would greatly benefit from a "map" of the underlying news stream to help identify and contextualize instances of adversarial activity within the larger volume of benign health discussion. Our evaluation of four different edge-cutting strategies over a 4k node, 200k edge graph generated from this dataset (Fig. 1, left) shows that the choice of strategy is subject to a variety of trade-offs. While the appropriate choice of strategy depends on the data, user, and task, all strategies significantly improved on the input "hairball" on standard and novel graph evaluation metrics.

We begin with a discussion of existing approaches to measuring graph qualities and making dense graphs usable, before presenting details of our approach and its application.

II. Related work

A. Interactive approaches to making dense graphs usable

Studies have shown that for graphs of 20-100 nodes, the orderable and overlap-free qualities of adjacency matrices help them to outperform node-link diagrams on common graph analysis tasks (except path finding) [15]. Interactive layout manipulation can improve the readability of node-link layouts both locally (e.g., magic lenses for separating [45] or shortening [32] links) and globally (e.g., through spatial group partitions [37] or topological layout [1]). Nodes can be aggregated to create meta-layouts with reduced complexity (e.g., by topological motif [9], node attribute [44], or node community [21]) or duplicated across multiple metric spaces with selective display of links within and between spaces (e.g., linked adjacency matrices [18], scatterplot matrices [3], parallel lists [14][43], and other "semantic substrates" [40]). Link volume can be managed through bottom-up node expansion (e.g., as trees [28] or linked subgraphs [23]) or eliminated through entirely link-free forms (e.g., zoomable adjacency matrices [11], fields [30], and B-matrices [22]). While these approaches solve some problems with large or dense node-link diagrams, they also introduce usability challenges in terms of an abstraction barrier (e.g., interpreting matrix cells as links), interaction burden (e.g., reordering matrix axes until structure emerges), or both.

B. Algorithmic approaches to making dense graphs usable

There are two main algorithmic approaches to graph reduction: graph aggregation and graph filtering [26]. In graph aggregation, nodes and edges are merged based on common attributes or attribute ranges to create a simplified structure. This process can be repeated recursively to create a hierarchical structure for interactive exploration (e.g., of nested communities [21]). In graph filtering, nodes and edges are removed either stochastically through representative sampling [29] or deterministically using metric-based heuristics. Past work also differentiates graph *sparsification*, which aims to filter edges while preserving graph properties within a given tolerance, from graph *pruning*, which aims to reveal core structures whose properties are obfuscated by noise [7]. The most common strategy for edge filtering is the removal of low-weight edges [2][46]. Another classic example is removing edges in decreasing order of betweenness centrality to isolate the communities of a graph [16]. The inverse heuristic, removing edges in increasing order of edge betweenness centrality, has been used to isolate the skeletal structure of pathways in scale-free communication networks [20]. Hybrid approaches have also been explored, such as filtering edges based on comparisons to a null model "ensemble" of random graphs resembling the input graph [7]. Since cutting edges from low-degree nodes can rapidly fragment a graph, post-processing can also be used to reconnect subgraphs into a single connected component [20].

C. Algorithmic approaches to measuring graph qualities

We are interested in the partitioning of graphs into substructures (i.e., groups, communities, clusters) of densely-connected nodes on both a structural and visual level. The quality of such a partition can be estimated using the modularity metric [16][34], which compares the strength of connections within and between communities to that of a random graph with equivalent connectivity. Louvain community detection [4] is a standard approach to discovering the partition that maximizes the modularity for a given graph. Recent work has shown that such community structure is resilient to deterministic edge filtering based on edge weight [46]. It has been speculated that edge filtering may produce more effective clusters for a variety of exploratory tasks including discovering unexpected nodes in a known cluster [41]. Graph aesthetics have also been proposed [36], converted into objective functions such as "stress" [25], and incorporated as readability metrics into graph tools [10].

D. Summary

Many graph representations developed to counter the scale issues of node-link diagrams were conceived in an era when "large" meant several hundred nodes, graph research was tackling the emergence of online social networks (e.g., [17][31][35]), and the primary concern was interactive exploration of multivariate, heterogeneous graphs. These representations cannot typically create usable and holistic overviews in the region of 1,000 to 100,000 nodes – a scale that is now common for graphs generated by NLP and ML techniques. Converting large graphs into drill-through hierarchies (e.g., [21]) is a promising approach in general but is not without its drawbacks. In particular, it loses the benefits of a single, flat layout that exposes community membership and provides a navigable and memorable frame of reference. This is particularly important for adversarial use cases where nodes and communities of interest could appear anywhere in the graph. Similarly, since node communities may only appear "odd" on inspection of their members rather than the attributes or graph metrics associated with those members, they may not easily be discovered in ranked representations like lists and adjacency matrices that emphasize extreme values.

Overall, node-link diagrams may still be the best choice for representing such graphs if we can create them with the appropriate map-like qualities that enable effective orientation, navigation, and revisitation.

III. Approach

An ideal edge cutting strategy would remove only those edges necessary for communities to clearly separate under force-directed layout, in ways that facilitate community-level tasks by graph users. Such tasks include identifying communities of structural or semantic interest, the key nodes within them, and the key links within and between them.

To optimize the structure of the graph for such tasks, we can perform edge cuts aiming to maximize the modularity of the filtered graph while preserving its inherent connectedness. However, it is not possible to know in advance which community semantics would best serve the user and their tasks. A range of possible community-level insights are as follows, indicating the extent to which prior knowledge, goals, and expectations play a role in making a graph usable:

1. *Unexpected community*. Discovering a community containing known nodes not expected to be interrelated.
2. *Unexpected membership*. Discovering an unexpected or unknown member in a known community of nodes.
3. *Unexpected connection*. Discovering an unexpected connection between known nodes across communities.
4. *Unexpected degree*. Discovering a known or unknown node with an unexpectedly high degree.

Since cutting edges from a graph influences not just the structure but the semantics of communities, different edge cutting strategies may achieve similar modularity scores for communities with fundamentally different characteristics. To provide users with a graph best suited to their analytic goals, it may be desirable to present them with a selection of "skeletal" graph summaries that clearly exhibit these alternative characteristics with very low edge-to-node ratios. Even such strongly filtered graphs may be sufficient for the community-level tasks listed above. They may also reveal the presence of "noise" nodes – both structurally and semantically – to be removed from the data before regenerating the graph.

A natural side-effect of minimizing total graph edges is that it also minimizes the number of inter-community connections that blur community boundaries and contribute to the hairball effect. "Filling in" skeletal communities with internal edges sourced from the original graph should therefore contribute shape and structure that was lost during the initial edge-cutting process. The communities detected in this filled-in structure should exhibit the same kind of visual concentration and separation as the regions on a geographic map when rendered using an appropriate force-layout algorithm. Sizing nodes according to their degree is another way of creating map-like landmarks that further support user orientation, navigation, and revisitation.

Converting hairball-like graphs into feature-rich, map-like graphs is central to our approach to making dense graphs usable, and in the following sections we expand on our two-stage method of generating and filling-in skeletal structures based on a range of edge cutting strategies. While we use Louvain [4] to partition graphs into communities, our approach generalizes to any community-detection method.

A. Graph definition

Our approach focuses on the class of weighted undirected graphs $G = \{V, E, w\}$ where $w : E \to \mathbb{R}_{\geq 0}$ is a weight function defined on graph edges. Our motivating use case is large-scale text analytics in which V represent named entities, E represent co-occurrence relations between entities in a document, and w represent the frequency of entity co-occurrence. However, the approach we present is not limited to this domain and should help to reveal structure in any large, dense, undirected graph with positive edge weights. For the purpose of community mapping, directed graphs can be converted into this format using any appropriate combination function (e.g., sum, mean, min, max). We also constrain the input graph to a single connected component, under the assumption that the multiple components of a disconnected graph can be processed individually and independently.

B. Edge cutting strategies

Our approach to graph reduction focuses on the filtering of edges while preserving the nodes and connectedness of the input graph. While nodes themselves may be introduced or duplicated as part of an errorful graph generation process, an appropriately filtered and rendered graph should help to reveal such noise without affecting the clarity of the overall structure. Similarly, maintaining the connectedness of the graph preserves paths between all pairs of nodes in ways that allow peripheral nodes to be reached via central node and vice versa.

Any node- or edge-level metric could form the basis of a deterministic edge cutting strategy, and any such strategy could be used within our overall approach. We focus on four:

Decreasing edge betweenness. Since removing edges in decreasing order of betweenness is an effective way to identify community structure in a graph that retains those edges [16], it follows that removing those edges will allow the resulting communities to separate when rendered as a node-link layout.

Increasing edge frequency. Filtering edges by frequency (or other manifestations of aggregate edge weight) is often performed during algorithmic graph pre-processing using a threshold for edge inclusion [2]. Some graph tools also allow such filtering to be performed interactively with immediate visual feedback on the graph layout.

Increasing edge information. The inclusion of high-degree nodes is one of the primary causes of the hairball effect. In the case of co-occurrence analysis, this occurs when entities occur frequently and independently of one another. Links between such entities are "noise" with low information value, i.e., knowing about the presence of one entity does not reduce uncertainty about the presence of the other. This is in contrast to pairs of entities that almost exclusively occur together, where knowing about the presence of either entity removes uncertainty about the other. Such edges have high information value, which can be captured by the information-theoretic concept of mutual information:

$$I(X;Y) = \sum_{x,y} p(x,y) \log\left(\frac{p(x,y)}{p(x)p(y)}\right)$$

where $p(x)$ is the probability of occurrence for entity x and $p(x,y)$ the probability of co-occurrence for entities x and y for the text analytics use case. This can be calculated directly from occurrence and co-occurrence counts (e.g., across a corpus of documents) or estimated from weighted edge lists by using the edge's weight in the calculation of $\hat{p}(x,y)$ and the sum of each linked node's edge weights in $\hat{p}(x)\hat{p}(y)$.

Random. The edge list is shuffled and converted into a priority queue for edge removal in random order.

C. Edge cutting process

For the given graph reduction strategy, edges are rank-ordered in a priority queue and removed sequentially up to a specified target if the degree of each linked node is greater than one (since cutting such edges will fragment the graph). We express edge cut targets as the number of edges per node desired in the filtered graph and continue attempting to remove edges in priority order until the target is reached. Since this process is not guaranteed to maintain global connectedness, we check whether the number of connected components in the filtered graph exceeds one. If this is the case, we iteratively reconnect components in decreasing order of edge removal priority until connectedness is restored.

D. Graph evaluation metrics

Our two independent parameters are the edge cutting strategy and the target density of the skeletal graph, expressed in terms of edges per node. We evaluate the resulting graphs using a range of structural, semantic, and visual metrics:

Edge count (structural). For skeletal graphs, this is a natural consequence of the target number of edges per node, although the final number can vary slightly because of the need to add edges to restore connectedness. For skeletal graphs with filled communities, it can vary widely based on the number of matching edges in the input graph.

Graph modularity (structural). This is a standard measure of community quality in the range [-1, 1] [16][34]. Empirically, modularity scores in the range [0.65, 0.85] for a single connected component generally provide well-defined communities with clear, uncluttered relations to one another.

Graph cohesion (structural). Excessive fragmentation of the graph in the filtering phase leads to weak connections between communities in the reconnected graph. We define the cohesion of a filtered graph to be the proportion of nodes remaining in the largest connected component after the target number of edges have been cut.

Community count (structural). Derived from the graph partition that maximizes modularity for the given graph structure. The optimum number of communities depends on the nature of the data, the user, and their task, but should not generally exceed the number that may be sequentially evaluated by a user engaged in community analysis tasks.

Degree assortativity (structural). Measures the Pearson correlation co-efficient between the degrees of linked nodes in the range [-1, 1] [33]. Positive values mean that nodes preferentially connect to nodes of a similar degree and are indicative of clique-like patterns. Negative values represent disassortative mixing between nodes at opposite ends of the degree spectrum and are indicative of hub-and-spoke patterns. While clique-like communities may be best for generating *unexpected member* insights, hub-and-spoke communities may be best for generating *unexpected degree* insights.

Type assortativity (semantic). Measures the Pearson correlation co-efficient between the types of linked nodes in the range [-1, 1] [33] in instances where nodes are heterogeneous or may be categorized according to their attributes. If nodes tend to connect based on their type, it may be easier to achieve *unexpected connection* insights.

Community homogeneity (semantic). The homogeneity of communities for graphs with heterogeneous nodes can be measured using the Simpson index [42], which gives the probability that two entities taken at random from the dataset of interest are of the same type. An overall score is calculated as the mean Simpson index across all communities. High homogeneity may support *unexpected community* insights.

Concentration stress (visual). To measure the extent to which communities of nodes visually cluster in the rendered layout, we adapted the stress model of graph layouts [25]. The original stress model is a cost function based on the sum of squared differences between graph-theoretic node distances and Euclidian node distances, used in the evaluation and iterative improvement of force-directed layouts. We define community concentration stress as the sum of squared differences between community node counts and the area of the convex hull bounding the community, normalized by the number of communities. Communities that are stretched or otherwise not concentrated in a single spatial region will increase this metric, as will variance in node density across the layout. Map-like layouts with uniform node density and clustered communities will have low concentration stress.

Separation stress (visual). One of the causes of the hairball effect is visual overlap between weakly separated communities. We define separation stress as the proportion of the graph area defined by its convex hull that contains a single community, with separation stress of zero indicating no overlap between communities and one indicating complete overlap. The non-overlapping area is calculated by iteratively taking the geometric difference between a community's convex hull and the convex hull of all other communities and summing the results across communities. Lower scores are better, although a small degree of overlap may be desirable to highlight stronger relationships between communities.

E. Layout generation process

While graph evaluation metrics can provide objective measure of graph qualities, it remains important to visually inspect and qualitatively compare actual graph layouts to assess their subjective usability for a given dataset and collection of tasks. We selected the ForceAtlas2 layout algorithm [19] for its free availability within Gephi [1] (a popular, open-source software tool for graph visualization) and ability to create uniform-density, circle-packed layouts for a range of competing graph structures. Such consistency allows for simple side-by-side comparisons between layouts and their community structures, especially when node color is used represent community membership. We generate small-multiples of graph layouts under a range of edge cutting strategies and target edges per node densities, for both skeletal and filled graphs, as a way of correlating these subjective judgements with our objective evaluation metrics.

F. Implementation

We have automated our approach using a combination of open-source and freely-available libraries in Python and Java. We use Python's `networkx`[2] to build and filter graphs, extract connected components, compute betweenness centrality for edge ranking, calculate assortativity metrics for graph evaluation, and export graphs into GDF files. Communities

[1] https://gephi.org/
[2] https://networkx.github.io/

are assigned as node attributes and the frequencies of retained edges as edge weights. We use `python-louvain`[3] for Louvain community extraction [4] and modularity scoring [16][34], incorporating edge weights, random node orderings, and a resolution of 0.8. For each graph partitioning, we select the highest modularity result from three runs. Our visual metrics use `SciPy`[4] for convex hull calculation and `Shapely`[5] for spatial analysis of community polygons.

We use the Gephi Toolkit[6] in Java to automate GDF graph ingestion, layout, styling, and export as both PNG images and GraphML layouts. GraphML files allow full interaction with pre-rendered graphs in Gephi and evaluation using our Python scripts. For graph layout, we use a random layout followed by 1500 iterations of ForceAtlas2 with approximate repulsion, a scaling factor of 1000, and Strong Gravity enabled. We map node sizes in the range [10,100] as a linear function of degree. These options reliably create the uniform-density, circle-packed layouts that admit small-multiple comparisons.

G. Summary

The approach presented in this section is designed to increase the usability of any dense graph when rendered as a node-link diagram. No edge-cutting strategy is universally optimal, the possible strategies are not limited to the four presented here, and users may find greater value in either the skeletal or filled layouts according to their needs. In the following section, we walk through our approach using an example graph and show how analysis of a representative graph in a given domain might suggest appropriate heuristics for automating the creation of candidate graphs in general.

IV. Case Study

Our example analysis is on media coverage of vaccines and the debate about their safety. This is a well-known target for communication that is both automated and adversarial [5], and it occurs on a scale that demands automated text analytics. This domain is also representative of text analytics in general.

A. Data collection and graph instantiation

We use a custom crawler to index a wide range of online news articles. From this index, we extracted 145k news articles published over the three-month period from January to March 2017 that contained the text "vaccine" or "vaccination". We then ran a custom named entity recognition engine over each article, identifying 300k unique entities across 20 generic types (e.g., Person, Organization, Concept, Location). Since there were over 250M edges connecting these entities based on document co-occurrence, we limited analysis to specific types of these entities as defined in a linked knowledge base. Around 180k of the generic entities mapped to the richer ontology of this linked knowledge base.

As a narrower topic of investigation, we focused on the relationship between media and medical entities related to vaccination discussion. We filtered our document set to only those documents containing entities with "vaccine" or "vaccination" in their name and our entity set to only those entities with a top-level medicine or media categorization. This resulted in 198,802 edges linking 3,973 entities, which themselves spanned four types (tv, radio, print, and medicine).

For systematic review of communities in this adversarial communication context, graph layouts should facilitate sequential evaluation of communities that are well-defined but not fragmented (medium-high modularity and cohesion), with clear boundaries (low separation stress) and uniform density (low concentration stress). Different patterns of degree- and type-assortativity and community size and homogeneity will enable different kinds of insight, e.g., discovering "vaccine injury", "vaccine failure", and "vaccine controversies" as related concepts in a single, clique-like community of similar medical entities, versus as hubs of independent communities connecting to the most closely entities of all types.

B. Analysis of the raw input graph

The rendering of the raw graph resulting from our graph instantiation process is shown in Fig. 1. It has all the characteristics of the hairball effect, including a high edge to node ratio, a scale-free, power-law degree distribution, and a low modularity ($Q = 0.26$) resulting from a small number ($N = 11$) of ill-defined and overlapping communities. Edges almost completely fill the area of the graph and adding labels to nodes would increase visual clutter even more. It is a good example of a large, dense graph generated by modern NLP and ML techniques that is unusable in its raw form.

C. Comparative evaluation of edge cutting strategies

The visual results of applying the three non-random edge-cutting strategies to our raw input "hairball" are shown in Fig. 2. The first high-level observation is that all strategies help to reveal skeletal structure in the graph that was otherwise hidden. The second is that very low edge-to-node ratios (1 or 1.5) may go too far, fragmenting structures that may have proved useful. The third is that clique-like communities appear to result from betweenness cuts and hub-and-spoke patterns from both frequency and information cuts, but with much fewer communities extracted for information cuts than the others. These subjective observations are confirmed by examining the evaluation metrics for each strategy (Fig. 3), averaged over the GDF and GraphML files from three runs.

Edge count. All strategies are very similar, with a higher count for low ratio betweenness cuts reflecting the extra edges required to reconnect a highly fragmented input graph.

Graph modularity. Drops immediately and rapidly for frequency, information, and random cuts as the edge-to-node ratio increases. It starts lower for betweenness cuts but quickly rises to a global maximum before gradually decreasing. All strategies increase the modularity of the raw input graph.

Graph cohesion. Rises quickly then plateaus for each strategy, with betweenness cuts almost completely shredding the input graph into many connected components for low edge-to-node ratios. Given that the elbows of all four curves occur at an edge-to-node ratio greater than the ratio yielding maximum modularity, this suggests using the elbow of the coherence curve to select a structurally-optimum edge-to-node ratio in the absence of overriding external factors.

Community homogeneity. All non-random strategies raise the homogeneity of communities above the input graph and the increase continues with greater numbers of edges per node. Random cuts reduce homogeneity below the input graph.

[3] https://github.com/taynaud/python-louvain
[4] https://github.com/scipy/scipy
[5] https://github.com/Toblerity/Shapely
[6] https://gephi.org/toolkit/

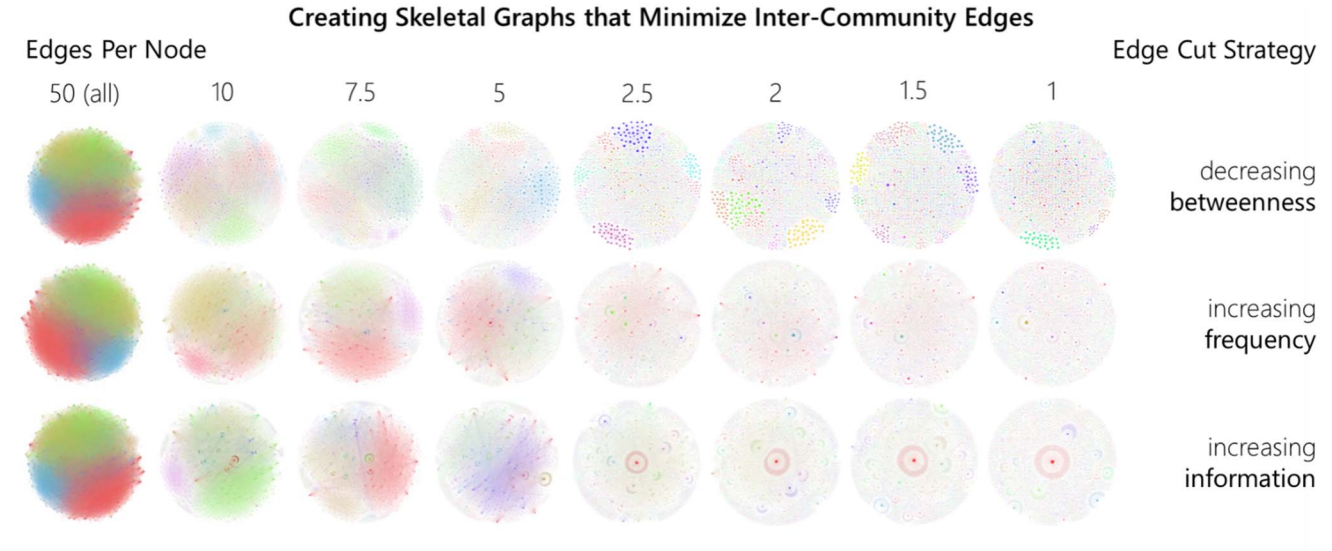

Fig. 2. How skeletal community structure emerges from an initial "hairball" graph for different edge-cutting strategies and target numbers of edges per node.

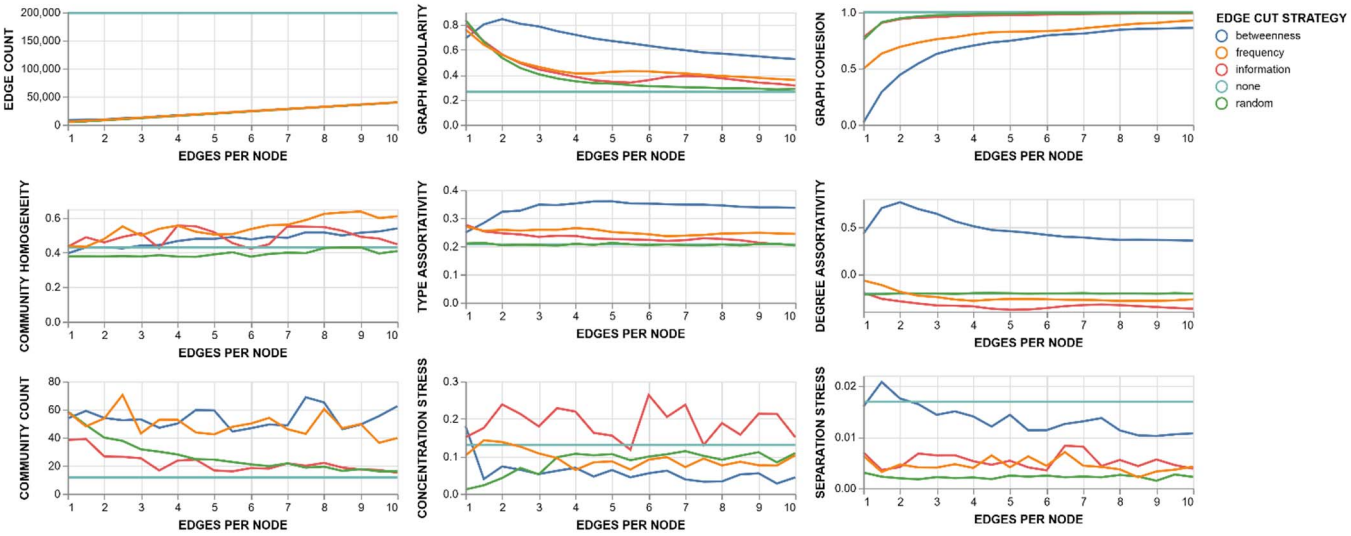

Fig. 3. How the emergence of structure is reflected by evaluation metrics and varies by edge cut strategy. The "none" reference lines characterize the raw graph.

Type assortativity. For all non-random strategies, nodes tend to connect to nodes of the same type to a greater extent than in the input graph. This effect is highly pronounced for betweenness cuts. Random cuts leave the metric unchanged.

Degree assortativity. Betweenness cuts create very high degree assortativity because they encourage clique-like communities with dense internal connections and weak bridges to other communities. Frequency and information cuts quickly reduce the degree assortativity below the baseline of the input graph, creating disassortative hub-and-spoke communities. This shows how alternative edge cut strategies can alter an input graph in opposing ways. Random cuts do not vary from the baseline metric of the input graph.

Community count. All non-random strategies are flat or gradually decreasing as the edges per node increase, with information cuts resulting in reliably fewer communities (as observed in the small multiples visualization of Fig. 2). Random cuts show a rapid reduction towards the baseline.

Concentration stress. Betweenness cuts rapidly drop concentration stress lower than the baseline graph. Frequency cuts require higher edges per node to drop similarly, while random cuts show the inverse trend. Information cuts, however, have substantially higher concentration stress than the baseline of the input graph. This is a consequence of ForceAtlas2 layout algorithm, which uses a degree-dependent repulsive force to guide layout. Since information cuts create clusters around high-degree hubs and ForceAtlas2 creates extra space around these hubs, the spatial density of nodes varies in ways that disrupt the relationship between community node counts and community area.

Separation stress. Frequency, information, and random cuts significantly reduce separation stress below the baseline. Betweenness cuts begin higher as a result of overlaps between weakly connected communities, but soon drop below as they result in better-defined communities throughout the graph.

We now repeat this analysis on the denser graphs that result from reinstating all edges from the input graph that are internal to communities in the skeletal graph (Fig 4., Fig. 5). This analysis reveals both similarities and differences to skeletal graphs. *Edge count* increases similarly with increasing edges per node, but with betweenness skeletons creating consistently lower final densities. *Graph modularity* starts high and remains high across all edges per node, with information cuts typically offering ideal modularity values in the range [0.65,0.85]. *Graph cohesion* is unchanged as it reflects properties of the skeletal rather than the filled graph. *Community count* and *Community homogeneity* follow the

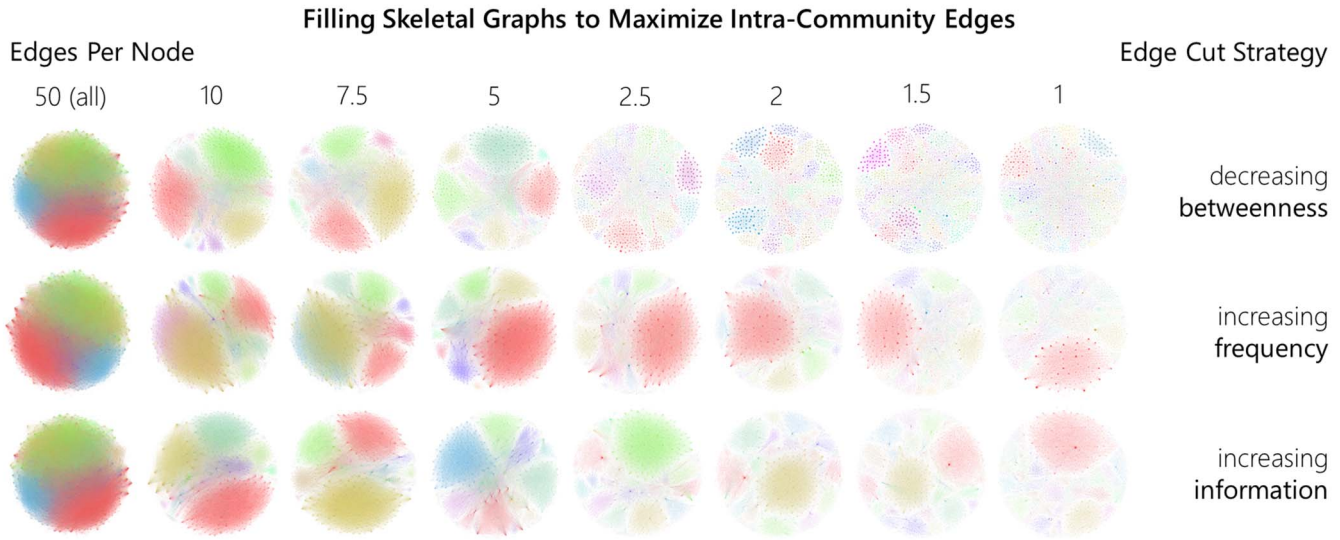

Fig. 4. How reinstating edges to skeletal communities creates dense, separated partitions for different edge-cutting strategies and target edges per node.

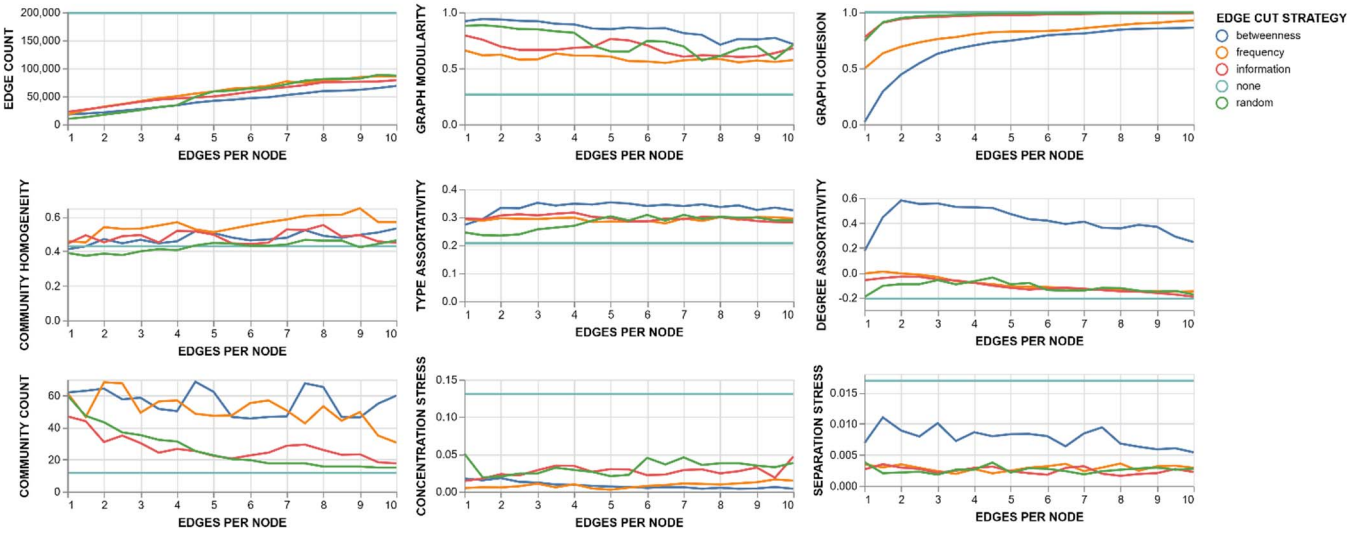

Fig. 5. How the reinstatement of internal community structure is reflected by evaluation metrics. The "none" reference lines characterize the raw graph.

skeletal graph results, but with frequency cuts creating communities with clearly higher homogeneity. *Type assortativity* is also similar, but with the random curve shifting above the input graph baseline. For *Degree assortativity*, all curves shift above the input graph baseline as a result of skeletal hub-and-spoke communities being "filled in". In terms of visual metrics, all strategies are well below the input graph baseline, accomplishing the primary goal of "trimming the hairball" into a usable structure. The relatively higher separation stress of betweenness layouts can be explained by the diffuse, fragmented nodes at the centers of layouts that do not form part of well-defined communities (Fig. 4). As with the evaluation of skeletal graphs, no one strategy is optimal across all metrics or community tasks. While random edge cuts produce the best structural results with high modularity and cohesion, betweenness cuts and information cuts produce communities with meaningful yet complementary structure and semantics. Frequency cuts generally offer intermediate results, but with the highest levels of community homogeneity and the lowest levels of graph modularity across the range.

Where specific use cases demand additional control over the density of intra-community edges, this may be achieved through a second round of edge cutting or the recursive application of our two-stage approach within communities.

V. Conclusion

The application of modern NLP and ML techniques to large-scale datasets can generate implicit graphs that are so densely connected as to be unusable when rendered as node-link diagrams. If such hairballs could be transformed into lower-density, map-like representations, they may better support user orientation, navigation, and revisitation across a variety of community-level tasks that are neither supported by existing tools nor recognized by existing taxonomies.

We have presented a two-stage approach to generating usable, map-like layouts from large, dense input graphs. This approach uses edge-cutting strategies based on node and edge metrics to first reduce a graph to a skeletal structure showing only essential relationships, before filling in the resulting communities to create dense but usable layouts. Each edge-cutting strategy has advantages and disadvantages, and the appropriate choice of strategy depends on the data, user, and task. In our case study, all strategies resulted in superior evaluation metrics above and beyond the input "hairball". More work is needed to evaluate this approach across a variety of input graphs and domains and to determine appropriate principles for automatically producing the most usable graph given a suitable specification of the intended tasks.

REFERENCES

[1] D. Archambault, T. Munzner, and D. Auber, "TopoLayout: multi-level graph layout by topological features", 2007. IEEE Transactions on Visualization and Computer Graphics, 13(2), 305-317.

[2] D. Auber, Y. Chiricota, F. Jourdan, and G. Melancon, "Multiscale visualization of small world networks", 2003. IEEE Symposium on Information Visualization, 75-81.

[3] A. Bezerianos, F. Chevalier, P. Dragicevic, N. Elmqvist, J.D. Fekete, "GraphDice: a system for exploring multivariate social networks", 2010. Computer Graphics Forum, 29(3), 863–872.

[4] V.D. Blondel, J.L. Guillaume, R. Lambiotte, and E. Lefebvre, "Fast unfolding of communities in large networks", 2008. Journal of statistical mechanics: theory and experiment, 10.

[5] D.A. Broniatowski et al., "Weaponized health communication: Twitter bots and Russian trolls amplify the vaccine debate", 2018 American Journal of Public Health, 108(10), 1378-1384.

[6] J. Danowski, "A network-based content analysis methodology for computer-mediated communication: an illustration with a computer bulletin board", 1982. In R. Bostrom (ed.), Communication Yearbook, 904-925. New Brunswick, NJ: Ransaction Books.

[7] N. Dianati, "Unwinding the hairball graph: pruning algorithms for weighted complex networks", 2016. Physical Review E, 93(1).

[8] J. Diesner and K.M. Carley, "Using network text analysis to detect the organizational structure of covert networks", 2004. In Proceedings of the North American Association for Computational Social and Organizational Science (NAACSOS) Conference, vol. 3.

[9] C. Dunne and B. Shneiderman, "Motif simplification: improving network visualization readability with fan, connector, and clique glyphs", 2013. In Proceedings of the ACM CHI Conference on Human Factors in Computing Systems , 3247-3256.

[10] C. Dunne and B. Shneiderman, "Improving graph drawing readability by incorporating readability metrics: a software tool for network analysts", 2009. University of Maryland, HCIL Tech Report HCIL-2009-13.

[11] N. Elmqvist, T.N. Do, H. Goodell, N. Henry, and J.D. Fekete, "ZAME: interactive large-scale graph visualization", 2008. In Proceedings of IEEE PacificVIS'08, 215-222.

[12] E. R. Gansner, Y. Hu, and S. G. Kobourov, "Visualizing graphs and clusters as maps", 2010. In IEEE Computer Graphics and Applications, 2259-2267.

[13] S. Ghani and N. Elmqvist, "Improving revisitation in graphs through static spatial features", 2011. In Graphic Interface, 737-743.

[14] S. Ghani, B.C. Kwon, S. Lee, J.S. Yi, N. Elmqvist, "Visual analytics for multimodal social network analysis: A design study with social scientists", 2013. IEEE Transactions on Visualization and Computer Graphics 19(12), 2032-2041.

[15] M. Ghoniem, J.D. Fekete, and P. Castagliola, "On the readability of graphs using node-link and matrix-based representations: a controlled experiment and statistical analysis", 2005. Information Visualization, 4(2), 114-135.

[16] M. Girvan and M.E.J. Newman, "Community structure in social and biological networks", 2002. Proceedings of the National Academy of Sciences, 99(12), 7821–7826.

[17] J. Heer and D. Boyd, "Vizster: visualizing online social networks", 2005. IEEE Symposium on Information Visualization (InfoVis), 5-12.

[18] N. Henry and J.D. Fekete, "MatLink: Enhanced matrix visualization for analyzing social networks", 2007. In IFIP Conference on Human-Computer Interaction, 288-302.

[19] M. Jacomy, T. Venturini, S. Heymann, and M. Bastian, "ForceAtlas2, a continuous graph layout algorithm for handy network visualization designed for the Gephi software", 2014. PloS one 9(6).

[20] Y. Jia, J. Hoberock, M. Garland, and J. Hart, "On the visualization of social and other scale-free networks", 2008. IEEE Transactions on Visualization and Computer Graphics, 14(6), 1285-1292.

[21] D. Jonker, S. Langevin, D. Giesbrecht, M. Crouch, and N. Kronenfeld, "Graph mapping: multi-scale community visualization of massive graph data", 2017. Information Visualization, 16(3), 190-204.

[22] S. Kairam, D. MacLean, M. Savva, and J. Heer, "GraphPrism: compact visualization of network structure", 2012. In Proceedings of the ACM Conference on Advanced Visual Interfaces, 498-505.

[23] S. Kairam, N. Henry Riche, S. Drucker, R. Fernandez, and J. Heer, "Refinery: visual exploration of large, heterogeneous networks through associative browsing", 2015. In Computer Graphics Forum, 34(3), 301-310.

[24] S.G. Kobourov, S. Pupyrev, and B. Saket, "Are crossings important for drawing large graphs?", 2014. In International Symposium on Graph Drawing, 234-245.

[25] Y. Koren and A. Çivril, "The binary stress model for graph drawing", 2008. In International Symposium on Graph Drawing, 193-205.

[26] T. Von Landesberger, A. Kuijper, T. Schreck, J. Kohlhammer, J.J. van Wijk, J.D. Fekete, and D.W. Fellner, "Visual analysis of large graphs: state-of-the-art and future research challenges", 2008. Computer Graphics Forum 30(6), 1719-1749.

[27] B. Lee, C. Plaisant, C. Sims Parr, J.D. Fekete, and N. Henry, "Task taxonomy for graph visualization", 2006. In Proc. ACM AVI, 1-5.

[28] B. Lee, C. Sims Parr, C. Plaisant, B.B. Bederson, V.D. Veksler, W.D. Gray, and C. Kotfila, "TreePlus: interactive exploration of networks with enhanced tree layouts", 2006. IEEE Transactions on Visualization and Computer Graphics, 12(6), 1414-1426.

[29] J. Leskovec and C. Faloutsos, "Sampling from large graphs", 2006. In Proceedings of the ACM KDD Conference on Knowledge Discovery and Data Mining, 631-636.

[30] R. Van Liere and W. De Leeuw, "GraphSplatting: visualizing graphs as continuous fields", 2003. IEEE Transactions on Visualization and Computer Graphics, 9(2), 206-212.

[31] A. Mislove, M. Marcon, K.P. Gummadi, P. Druschel, and B. Bhattacharjee, "Measurement and analysis of online social networks", 2007. In Proceedings of the ACM SIGCOMM conference on Internet measurement, 29-42.

[32] T. Moscovich, F. Chevalier, N. Henry, E. Pietriga, and J.D. Fekete, "Topology-aware navigation in large networks", 2009. In Proceedings of the ACM CHI Conference on Human Factors in Computing Systems, 2319-2328.

[33] M.E.J. Newman, "Mixing patterns in networks", 2003. Physical Review E, 67 026126.

[34] M.E.J. Newman, "Modularity and Community Structure in Networks", 2006. Proc. Natl. Acad. Sci. USA, 103, 8577–8582.

[35] A. Perer and B. Shneiderman, "Balancing systematic and flexible exploration of social networks", 2006. IEEE Transactions on Visualization and Computer Graphics, 12(5), 693-700.

[36] H.C. Purchase, "Which aesthetic has the greatest effect on human understanding?", 1997. In International Symposium on Graph Drawing, 248-261.

[37] E.M. Rodrigues, N. Milic-Frayling, M. Smith, B. Shneiderman, and D. Hansen, "Group-in-a-Box layout for multi-faceted analysis of communities", 2011. In IEEE Conference on Social Computing, 354-361.

[38] B. Saket, C. Scheidegger, S.G. Kobourov, and K. Börner, "Map-based visualizations increase recall accuracy of data", 2015. In Computer Graphics Forum, 34(3), 441-450.

[39] B. Saket, P. Simonetto, S. Kobourov, and K. Börner, "Node, node-link, and node-link-group diagrams: an evaluation", 2014. IEEE Trans. on Visualization and Computer Graphics, 20(12), 2231-2240.

[40] B. Shneiderman and A. Aris, "Network visualization by semantic substrates", 2006. In IEEE Transactions on Visualization and Computer Graphics, 12(5), 733-740.

[41] B. Shneiderman and C. Dunne, "Interactive network exploration to derive insights: filtering, clustering, grouping, and simplification", 2012. In Springer International Symposium on Graph Drawing, 2-18.

[42] E.H. Simpson, "Measurement of diversity", 1949. Nature, 163, 688.

[43] J. Stasko, C. Görg, and Z. Liu, "Jigsaw: supporting investigative analysis through interactive visualization", 2008. Information visualization, 7(2). 118-132.

[44] M. Wattenberg, "Visual exploration of multivariate graphs", 2006. In Proceedings of the ACM CHI Conference on Human Factors in computing systems, 811-819.

[45] N. Wong, S. Carpendale, and S. Greenberg, "EdgeLens: An interactive method for managing edge congestion in graphs", 2003. In IEEE Symposium on Information Visualization, 51-58.

[46] X. Yan, L.G.S Jeub, A. Flammini, F. Radicchi, and S. Fortunato, "Weight Thresholding on Complex Networks", 2018. arXiv preprint, arXiv:1806.07479.