Visual Informatics 4 (2020) 43–57

Contents lists available at ScienceDirect

Visual Informatics

journal homepage: www.elsevier.com/locate/visinf

OnionGraph: Hierarchical topology+attribute multivariate network visualization

Lei Shi [a], Qi Liao [b], Hanghang Tong [c], Yifan Hu [d], Chaoli Wang [e], Chuang Lin [f], Weihong Qian [g,*]

[a] ACT&BDBC, Department of Computer Science & Engineering, Beihang University, Beijing 100191, China
[b] Department of Computer Science, Central Michigan University, Mount Pleasant, MI 48859, United States
[c] Department of Computer Science, University of Illinois at Urbana-Champaign, Champaign, IL 61820, United States
[d] Yahoo Labs, New York, NY 10036, United States
[e] Department of Computer Science & Engineering, University of Notre Dame, Notre Dame, IN 46556, United States
[f] Department of Computer Science and Technology, Tsinghua University, Beijing 100084, China
[g] Alibaba Cloud, Beijing 100102, China

ARTICLE INFO

Article history:
Received 25 December 2019
Accepted 23 January 2020
Available online 11 February 2020

Keywords:
Multivariate network visualization
Hierarchical abstraction
Focus+context
Entropy

ABSTRACT

Hierarchical abstraction is a scalable strategy to deal with large networks. Existing visualization methods have allowed to aggregate the network nodes into hierarchies based on the node attributes or network topology, each of which has its own advantage. Very few previous system has the capability to enjoy the best of both worlds. This paper presents OnionGraph, an integrated framework for the exploratory visual analysis of heterogeneous multivariate networks. OnionGraph allows nodes to be aggregated based on either node attributes, topology, or a hierarchical combination of both. These aggregations can be split, merged and filtered under the focus+context interaction model, or automatically traversed by the information-theoretic navigation method. Node aggregations that contain subsets of nodes are displayed by the onion metaphor, indicating the level and details of the abstraction. We have evaluated the OnionGraph tool in three real-world cases. Performance experiments demonstrate that on a commodity desktop, our method can scale to million-node networks while preserving the interactivity for analysis.

1. Introduction

Information networks are intensively studied nowadays and many of them are *multivariate* in that their nodes and edges are associated with multiple variables. For example, in a co-authorship network, each node (the author) has its affiliation and research interest information, each edge (the co-authorship) has the collaboration date and a frequency. They are known as the node/edge attributes. In this paper, we consider an advanced version of the multivariate network, that further mixes up nodes/edges of different types, also known as the heterogeneous network. In the bibliographic network, the nodes can be an author from a university, a paper on certain topic, or a venue (i.e., conference/journal) happened in a location. The edge can represent the relationship of citation, authorization, etc. Each type of node/edge has its own set of attributes. Analyzing the heterogeneous multivariate network can lead to more insights than its homogeneous univariate counterpart. Besides knowing the authors in the center of a co-authorship network, it is also possible to detect the authors with highly cited papers at prestigious venues.

Visualizing heterogeneous multivariate networks is technically challenging. First, bringing multiple types of information together in real-world scenarios makes the underlying network large and complex. To allow users to perceive the overview of the network, there is the **summarization** problem: how to create the visual abstraction of a large heterogeneous network with both topology and attribute information? The popular multi-scale visualizations by the hierarchical graph clustering (Auber et al., 2003; Abello et al., 2006) serve large-scale networks well, but do not consider the additional node/edge attributes. Second, the multivariate nature of the network prohibits the visualization of all details of the topology and attribute information in the same picture. There is the **navigation** problem: which interaction model to apply to guide users from an initial high-level visual abstraction to detecting and analyzing the low-level network

* Corresponding author.
E-mail addresses: leishi@buaa.edu.cn (L. Shi), qi.liao@cmich.edu (Q. Liao), htong@illinois.edu (H. Tong), yifanhu@yahoo.com (Y. Hu), chaoli.wang@nd.edu (C. Wang), chlin@tsinghua.edu.cn (C. Lin), weihong.qwh@alibaba-inc.com (W. Qian).

Peer review under responsibility of Zhejiang University and Zhejiang University Press.

https://doi.org/10.1016/j.visinf.2020.01.002

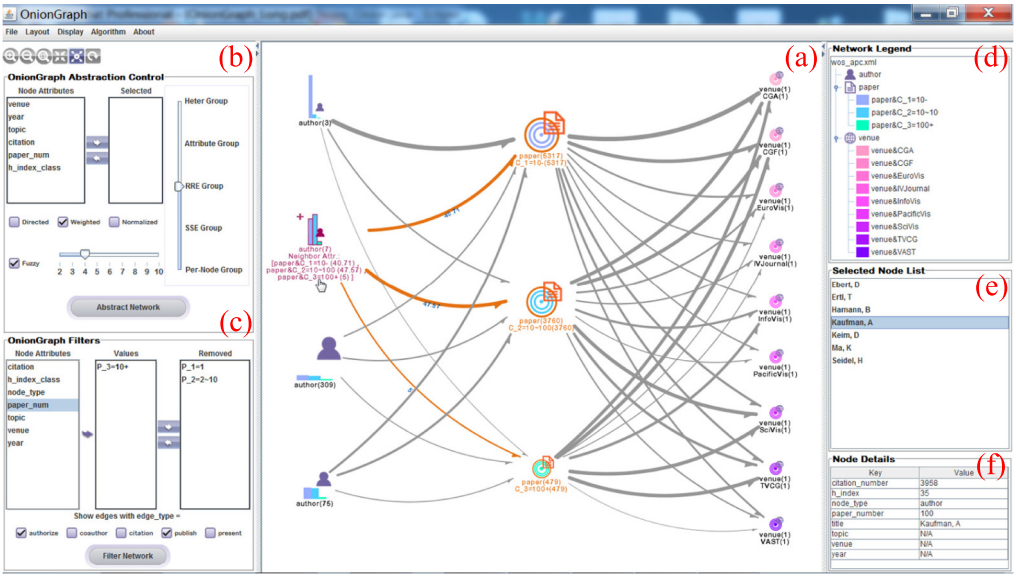


Fig. 1. OnionGraph interface showing the bibliographic network in the visualization community. (a) Main OnionGraph panel visualizing the author-paper-venue network. Venues (right column) are expanded into single journal/conference entities; papers (central column) are expanded by their citation counts; authors (left column) are expanded by their neighborhood attributes, i.e., the publication profile of low/medium/high-citation papers. The layout improves PivotGraph grid-like layout (Wattenberg, 2006). (b) Abstraction control panel. The current abstraction profile is applied on the selected author group in the main panel. (c) Filter panel configured to only show authors who publish more than 10 papers, and edges which connect author-paper ("authorize" type) and paper-venue ("publish" type). (d) Legend panel showing the icons used in the current abstraction. (e) The selected node (author) list. (f) Details of "Kaufman, A", all the statistics (e.g., h-index) are computed within the visualization community.

and its attribute pattern? For example, in the multi-scale network visualization, the hierarchy-traversing model is applied for exploring the network structure.

Despite a wealth of literature in the network visualization research (Herman et al., 2000; Battista et al., 1998), only a few are designed for multivariate networks. PivotGraph (Wattenberg, 2006) and OntoVis (Shen et al., 2006) are early works addressing such needs. They generate visual abstractions using either the network topology or node attributes, but not both. Moreover, they focus on the static abstraction, but do not support the interactive network exploration. Essentially, existing visualization methods fail to deal with heterogeneous multivariate networks. First, the algorithm to summarize the network based on either topology or attribute information alone will not produce consistent node clusters. For example, by topology-based graph clusterings, each cluster indicates a group of nodes with denser internal connections than external ones. Nevertheless, these dense node groups can have rather diversified internal attribute distributions. Second, the standard hierarchy-traversing interaction model works well on topology-based clusterings where each cluster has a self-contained local structure. On heterogeneous networks, the hierarchy-traversing interaction may not be appropriate, as the cross-cluster connections are sometimes more important than the internal ones.

In this paper, we present OnionGraph,[1] an integrated framework for the exploratory visual analysis of large, multivariate and heterogeneous networks. Fig. 1 gives an overview to the OnionGraph visualization interface. The main panel in Fig. 1(a) depicts a sample OnionGraph network abstraction. Each node group in the view is associated with an abstraction profile in five possible hierarchies ranging from the top/coarsest-level heterogeneous abstraction to the bottom/finest-level per-node granularity. This profile is customized by users through the abstraction control panel (Fig. 1(b)). The network abstraction can be further simplified by node/edge attribute filters (Fig. 1(c)), and the interesting part of the network is enlarged for analysis.

This work makes three contributions. First, we invent the hierarchical topology+attribute abstraction on heterogeneous multivariate networks, and develop the focus+context interaction model in navigating these abstractions (Section 3). Users can micro-manage the abstraction profile on each group of nodes to generate a fully customized network visualization, which can potentially reveal novel patterns. Second, we propose a suite of network partition/clustering algorithms organized in a top-down manner to generate such hierarchies in the abstraction (Section 4). These algorithms explicitly combine the topology and attribute information while guaranteeing a finer granularity as the user drills down to a lower hierarchy. Third, we introduce the "onion" visual metaphor to naturally represent the node group in the resulting network abstraction (Section 6). Notably, we design an information-theoretic framework to guide users in navigating the OnionGraph abstraction (Section 5). Our framework is evaluated in three real-world information networks (Section 7). All results demonstrate the effectiveness of OnionGraph in exploratory tasks over large, heterogeneous and multivariate networks.

2. Related work

This section summarizes the literature on multivariate network visualization. While there are a few surveys on this topic very recently, e.g., the book by Kerren et al. (2014) and the STARs paper by Hadlak et al. (2015), we take a different view by classifying the literature according to the classical InfoVis reference model (Card et al., 1999). As shown in Fig. 2, it is found that most related work can be categorized into five processing stages. There is only one modification to the original InfoVis model after incorporating the concept in the data state model (Chi, 2000): in the first stage, we separate the data transformation serving for the network generation from the visualization transformation which simplifies the network for the effective visualizations.

[1] The "onion" notation is also used in Sindre et al. (1993). However, both the visual metaphor and the application are significantly different.

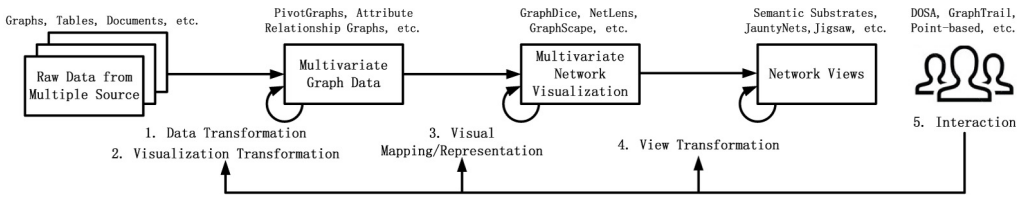


Fig. 2. Taxonomy of multivariate network visualization literature by the InfoVis reference model.

2.1. Data transformation

The first class of techniques cover the algorithm and framework on the *data transformation* from multiple raw data sources to multivariate graphs and networks, which our OnionGraph framework can take as the input. The Ploceus system by Liu et al. (2011) proposed an interactive interface for this purpose. It helps to construct networks over the tabular data for visual analysis. A comprehensive set of elemental operations are defined in Ploceus, consisting of both the low-level network generations and the high-level visualization transformations. The Orion system by Heer and Perer (2011) is similar to Ploceus in that it also generates networks from raw data tables. Compared with Ploceus that relies on the human supervision in the process, Orion supports semi-supervised network construction by automatically computing feasible network paths from a linking graph. In another approach of the attribute relationship graph by Weaver (2010), the construction of multivariate graphs is further combined with cross-filtered views to enable the effective discovery and dissection of the relationship data. The NetLens interface by Kang et al. (2007), though not designed to explicitly display relationships, introduced the content-actor model that captures the essence of many real-life multivariate networks.

2.2. Visualization transformation

Same with the data transformation, the output of the *visualization transformation* is again the multivariate graph. The major difference lies in that the input of this stage is already the graph data, and the goal is to simplify the large and complex graph for effective visualization. The idea of abstracting network topologies has been intensively studied in the literature, by applying graph clustering (Quigley and Eades, 2000; Abello et al., 2006), motif simplification (Dunne and Shneiderman, 2013) or graph compression techniques (Shi et al., 2013; Dinkla et al., 2012) (see the survey in Elmqvist and Fekete (2010)). On the attribute-centric transformation over multivariate networks, Wattenberg (2006) pioneered PivotGraph. It leverages a roll-up operation to pivot the nodes with the same value on one or two user-selected attributes into node aggregations. In another data selection operation, the network can be reduced to only node aggregations with specified attribute values. OntoVis by Shen et al. (2006) proposed the method of semantic and structural network abstraction over the ontology graph of heterogeneous social networks. On the attribute analysis, the network is filtered by selected nodes in the ontology graph. On the structural abstraction, OntoVis provides methods such as degree-one node and duplicate path reductions.

The OnionGraph framework shares the similar idea to abstract multivariate networks by node attributes. Beyond the singular attribute-centric aggregation in PivotGraph and the separate semantic/structural abstraction in OntoVis, OnionGraph allows the node aggregation by a combination of the topology and attribute information.

2.3. Visual mapping and representation

By comparing with the popular node-link visual metaphor design, we classify the visual representation method of multivariate networks into three classes: the singular node-link design, the hybrid approach combining the node-link graph with non-graph visual metaphors, and the non-graph design. For the singular node-link design, the additional node/edge attributes are either encoded into separate visual channels (Wattenberg, 2006; Tominski et al., 2009; Auber, 2004;node size, color, link thickness, etc.), or they are revealed by the node placement via the attribute-driven graph layout (discussed later in the view transformation stage).

On the hybrid approach, GraphDice by Bezerianos et al. (2010) introduced the scatterplot matrix design in visualizing multivariate social networks. Each node attribute on the network behaves as one row/column in the scatterplot matrix. The correlation between each pair of attributes is shown in the intersection of the matrix, where a node-link graph of the whole network is replicated. On multivariate state transition graphs, Pretorius and Van Wijk invented the bar tree to visualize both the clustering hierarchy and the metric data on graph nodes (Pretorius and van Wijk, 2006). Later, they introduced the parallel coordinates like visual metaphor in Pretorius and van Wijk (2008). The source and target nodes are aligned on two parallel coordinates which are clustered hierarchically according to node attributes. The GraphScape method by Xu et al. (2007) superimposed a landscape metaphor over the 2D node-link graph to visualize multivariate networks. The additional node attribute is displayed by the surface height in the third spatial dimension. FacetAtlas by Cao et al. (2010) overlays a contour map on the node-link representation to display the local context of multivariate networks.

Another sub-class of the hybrid approach juxtaposes the node-link graph with other visual metaphors in coordinated multiple views. Interactive visual queries and filters are often enabled for the iterative analysis over multivariate networks. For example in the attribute relationship graph (Weaver, 2010), cross-filtered views including the node/edge/attribute lists and the attribute correlation scatterplot, are displayed side by side, with the attribute relationship graph shown in the main view. In Ko et al. (2014), the node-link graph is overlaid on a geographical map and coupled this visualization with several coordinated views to display the associated spatial, temporal and contextual information.

The last class of methods replace the node-link graph design with other visual metaphors. ZAME by Elmqvist et al. (2008) proposed the adjacency matrix visualization through the use of multi-scale data aggregation. Edge attribute statistics on the aggregation are displayed on matrix tiles by eight types of glyphs designed for different tasks. NetLens (Kang et al., 2007) creates a series of statistical charts (e.g., bar charts) upon elemental attribute queries to serve the attribute-centric analytical tasks over multivariate networks. GraphTrail by Dunne et al. (2012) exhibits the similar statistical chart design to illustrate node/edge attributes. By linking sequential network attribute views into a trail, it enables the user to surf within the analysis history.

2.4. View transformation

View transformations in the InfoVis reference model include three sub-classes of methods: location probes, viewpoint controls and distortions. On multivariate networks, the first sub-class of the location probe is the most popular, which uses graph layouts to reveal the attribute information on networks. This is referred to as attribute-driven layouts.

PivotGraph (Wattenberg, 2006) proposed the grid-based layout which places each node by its value on one or two selected node attributes. The parallel coordinates metaphor can be viewed as an extension of this grid-based layout in that one coordinate is determined by the node type and another by the sorting attribute on each node type. Typical examples include Jigsaw (Stasko et al., 2008), PivotPaths (Dork et al., 2012), and state transition graphs (Pretorius and van Wijk, 2008). Semantic Substrates by Shneiderman and Aris (2006) introduced the user-defined layout by placing all the nodes into several non-overlapping regions according to one specified categorical node attribute. Essentially, many other types of node attribute, beyond numerical and categorical ones, can be used to drive the graph layout, such as the geographical information (van den Elzen and van Wijk, 2014).

Some attribute-driven layouts further incorporate the network structure to preserve the topology to some extent. GraphScape (Xu et al., 2007) modified the original spring-electrical force-directed model to reveal the attribute affinity on graph nodes. Wu and Takatsuka (2006) visualized multivariate networks on the surface of a sphere by the 3D self-organizing map. JauntyNets (Jusufi et al., 2013) creates attribute nodes and places them in the circle around the main graph view. The nodes with attribute value above certain threshold are linked with the corresponding attribute node to plug-in the attribute information into the topology-based layout.

2.5. Interaction

Interaction controls are often integrated into the processing stages mentioned above. On the data transformation, PivotGraph (Wattenberg, 2006) allows users to pick a pair of node attributes to roll-up and to query by node attribute values for the sub-graph selection. On manipulating graph views, the latest interaction technique from Detail to Overview via Selection and Aggregation (DOSA) (van den Elzen and van Wijk, 2014) proposed the method to directly brush and/or select on the attribute-driven graph layout, and then aggregate the selected nodes by the PivotGraph-like transformation. On visualizing network attributes, NetLens (Kang et al., 2007) offers interactions to iteratively refine the query on multivariate networks. GraphTrail (Dunne et al., 2012) integrates the multiple chart design with the drag-and-drop interaction to capture the user's exploration history.

Another thread of relevant interactions target at manipulating network hierarchies, mostly the hierarchy navigation and editing. Elmqvist and Fekete (2010) classified the hierarchical aggregation based visualization into five types: above traversal, below traversal, level traversal, range traversal and unbalanced traversal. The navigation methods generally work to change the hierarchy setting within each type of the classification or switch between two different types. Auber et al. (2003) proposed the method to start from an above traversal and leverage an overview+detail navigation to create a below/range traversal. ASK-GraphView (Abello et al., 2006) allows the user to click on each node aggregation to expand under any traversal type and finally generates an unbalanced traversal. Topological Fisheye (Gansner et al., 2004) enables an interactive switching among unbalanced traversals by specifying focuses on the network. GrouseFlock (Archambault et al., 2008) provides high-level hierarchy modification operators based on the low-level delete and merge operations.

3. OnionGraph Framework

3.1. Principle

Hierarchical Topology+Attribute Abstraction. As mentioned, neither the attribute-based nor the topology-based network visualization method alone can serve the exploratory analysis tasks such as "Is there any VAST paper heavily cited by both TVCG and CGF papers?". Moreover, a flat combination of these two methods leads to fragmented network abstractions. For example, partitioning a social network according to both the user's community and their profile generates too many tiny clusters to be interpretable. In OnionGraph, we introduce the hierarchical topology+attribute abstraction principle. In high-levels, the original large multivariate network is aggregated by the semantic information (the node type and attributes). Interesting part of this network abstraction can be further drilled down into lower levels by exploiting topological features, both interactively and in-situ on the same abstraction view. The full picture of the abstraction is illustrated in Fig. 3 and more details are given in Sections 3.2 and 4.

Such a hierarchical design achieves the level-of-detail viewing on multivariate networks in that each lower-level hierarchy presents significantly more network details than its parent hierarchy. This is also why the semantic aggregation (level-I/II) is placed on top of topological methods (level-III/IV) in the design of the five-level hierarchy. With the appropriate node attribute selection and an optional binning operation, semantic aggregation can always create a compact abstraction of the entire network. On the other hand, most real-world networks have limited topology redundancy. Compressing these networks by topology-only methods leads to yet another cluttered network.

Focus+Context Exploration. State-of-the-art hierarchical network visualization methods assume a pre-computed hierarchy either by algorithms or by the inherent data organization. User explorations are limited to traversing fixed trails according to the hierarchy. This places many constraints on the analytics capability. Moreover, the network hierarchies, such as those by graph clustering, are sensitive to the parameters applied. Users can hardly understand why some parts of the network are grouped together.

In contrast, OnionGraph features the user-defined focus+context visual exploration design. First, we apply well-defined network abstraction algorithms by the topology+attribute principle. They generate network partitions without any unambiguity. Users are guided by algorithm heuristics, so that they can understand the output in exploring each network hierarchy. Second, multiple exploration steps can be spliced on the fly, then users can define the analysis flow and generate the customized view on demand according to different tasks and network characteristics. For example in Fig. 3, after a few steps of user navigation, a stratified network view having different levels of abstraction is created on demand to serve the complicated multivariate network analysis tasks.

Local Refinement + Global Filtering. In the OnionGraph exploration, each higher-level node is expanded in-situ into lower-level sub-nodes, leading to a local refinement approach. Following the visual information seeking mantra (Shneiderman, 1996), OnionGraph also implements attribute filters on network nodes/edges to let the user focus on important information. In the straightforward design, a separate filter can be attached to the profile of each local refinement, however, in reality users can hardly remember the detail of each filter. Therefore, it is hard to restore the network that is filtered entirely, as the filter's setting only governs the local network and cannot be accessed for changes. Finally, we adopt the global filtering mechanism which applies the same filter on the entire network. The filtered network is subsequently abstracted according to OnionGraph settings.

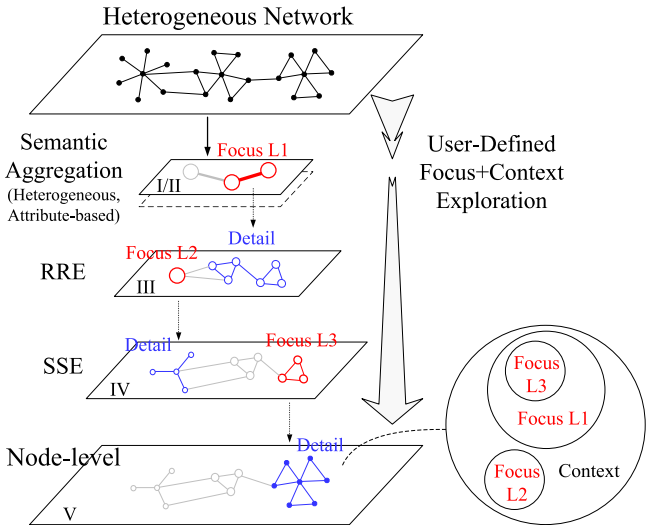


Fig. 3. OnionGraph data structure featuring five hierarchies below the original network: networks by semantic aggregations (SA) on node type (heterogeneous abstraction) and node attributes, Relative Regular Equivalence (RRE), Strong Structural Equivalence (SSE), and the node-level network in the finest granularity. In each hierarchy, the network can be expanded on certain focuses (red regions) into their lower-hierarchy details (blue regions). (For interpretation of the references to color in this figure legend, the reader is referred to the web version of this article.)

3.2. Data structure

Fig. 3 shows the five-level hierarchy of OnionGraph: the semantic aggregations (SA) in level-I/II are semantic, the level-III Relative Regular Equivalence (RRE) combines semantic and topological information, the level-IV Strong Structural Equivalence (SSE) is mostly topological.

In more details, the level-I abstraction groups the original network by the node type, level-II abstractions consider the categorical/nominal node attributes. Once a set of attributes are selected, the network is aggregated by grouping all the nodes with the same value on these attributes together. The network links are formed accordingly. In level-III, the RRE method extracts role sub-groups from the higher-level semantic aggregation. The role in the network is defined recursively in that, the nodes with the same set of roles in their neighborhood are considered having the same network role. The initial role partition is constructed through the semantic aggregation. The SSE method in level-IV is similar to RRE, except that the role definition is different. It defines the nodes having exactly the same set of neighborhoods to preserve the same role. The SSE role definition is stricter than that of RRE which only considers the role of neighborhoods. In the finest node-level (level-V), each SSE node group is split into individual network nodes, rolling back to the input network granularity.

3.3. Hierarchical navigation

An example of the OnionGraph navigation trail is shown in the bottom-right part of Fig. 3. By the interactive exploration, users can create multiple, hierarchical focuses over the network abstraction. Each focused sub-network is associated with an independent user-defined abstraction profile. The profile specifies both the current network hierarchy and the abstraction setting. In the OnionGraph visualization, the focused sub-network is expanded in-situ by the focus+context principle. The network view in the middle can juxtapose sub-networks with multiple hierarchy settings. This is quite different from the overview+detail network visualization and the hierarchy-traversing navigation that do not preserve context. More details on the actual user interaction are introduced in Section 6.2.

4. Algorithm

We implement the OnionGraph abstraction through a suite of algorithms that partition the network into node groups in multiple levels. To describe these algorithms, we first introduce the notations used throughout this section.

Heterogeneous Multivariate Network. Let $G = (V, E)$ be a directed and weighted network, where $V = \{v_1, \ldots, v_n\}$ and $E = \{e_1, \ldots, e_m\}$ denote the node and link sets. Define W as the adjacency matrix where w_{ij} denotes the link weight and $w_{ij} > 0$ indicates a link from v_i to v_j. On each node v_i, let $N^+(v_i) = \{j | w_{ij} > 0\}$ and $N^-(v_i) = \{j | w_{ji} > 0\}$ be its outbound and inbound neighborhood set, both representing the node's connection pattern. Let $D = \{d_1, \ldots, d_s\}$ be the type and attributes of the network nodes in G, with s dimensions in total. $D(v_i) = \{d_1(v_i), \ldots, d_s(v_i)\}$ denotes the type/attribute vector of the node v_i, where $d_k(v_i)$ is the value on the kth dimension.

Network Partition. Let $P : V \to \{1, 2, \ldots, t\}$ be a partition function (role assignment, coloration, or grouping interchangeably) of the network G into t groups of nodes. $P(v_i)$ returns the group index of node v_i after the partition. The algorithms to create the OnionGraph abstraction are equivalently defined by the network partition achieved. Below we describe these partition functions, their actual implementation, as well as the running performance.

4.1. Semantic aggregation

Semantic Aggregation (SA) creates partitions of the network by the selected set of node type or attributes. Formally, for any nodes v_i and v_j in a network G, given the selected attribute set $\overline{D} \subseteq D$, the semantic aggregation network partition P satisfies:

$$P(v_i) = P(v_j) \Leftrightarrow \overline{D}(v_i) = \overline{D}(v_j) \tag{1}$$

Fig. 4(a) illustrates such a partition based on the node attribute having values "I" or "II".

In OnionGraph structure (Fig. 3), the level-I abstraction is by SA partition using the node type. For example in Fig. 9(a), an initial OnionGraph view of the bibliographic network shows three node-type groups, i.e., papers, authors and venues. The level-II abstraction also follows the SA partition, but works over the user-selected node attributes. Multiple network hierarchies can be created when these attributes are applied sequentially.

4.2. Relative regular equivalence

The original regular equivalence concept is defined recursively on the network node by the same set of neighborhood roles (White and Reitz, 1983). For any nodes v_i and v_j in a network G, a regular equivalence network partition P satisfies:

$$P(v_i) = P(v_j) \Rightarrow P(N^+(v_i)) = P(N^+(v_j)) \textit{and } P(N^-(v_i)) = P(N^-(v_j))$$

However, directly applying the regular equivalence on a network leads to many possible partitions. Fig. 4(b) gives a particular case. In the extreme, the identity partition (each node has a different role) and the complete partition (each node has the same role) are both regular. In OnionGraph, motivated by the hierarchical design, we propose a practical solution to apply Relative Regular Equivalence (RRE) on top of the existing SA partition. Formally, the RRE partition P over a SA partition P_0 satisfies:

$$\begin{aligned} &P(v_i) = P(v_j) \Leftrightarrow P_0(v_i) = P_0(v_j) \textit{ and} \\ &P_0(N^+(v_i)) = P_0(N^+(v_j)) \textit{ and } P_0(N^-(v_i)) = P_0(N^-(v_j)) \end{aligned} \tag{2}$$

Fig. 4(c) gives an example of the RRE partition relative to the SA partition in Fig. 4(a).

4.3. Strong structural equivalence

More stringent to the regular equivalence, Strong Structural Equivalence (SSE) partition (Lorrain and White, 1971) requires the network nodes to have exactly the same neighborhood set. For any nodes v_i and v_j in a network G, the SSE network partition P over a RRE partition P_0 satisfies:

$$\begin{aligned} &P(v_i) = P(v_j) \Leftrightarrow P_0(v_i) = P_0(v_j) \ \ and \\ &N^+(v_i) = N^+(v_j) \ \ and \ \ N^-(v_i) = N^-(v_j) \end{aligned} \tag{3}$$

Besides the above definition on the directed networks, there are other variations of the RRE/SSE partition. The undirected RRE/SSE (Fig. 4) considers the union of inbound and outbound neighborhood sets, the weighted RRE/SSE considers the number of neighborhood role occurrences (RRE) and the weight of the connecting edges (SSE). These options are included in the OnionGraph design and configurable by the user.

4.4. Fuzzy equivalence

In many real-world cases, the strict RRE/SSE partition leads to an excessive number of groups as these networks are too complex to have strict structural equivalences. We introduce the fuzzy RRE/SSE partition which allows users to control the number of partitions in the abstraction.

The first step is to represent each node v_i by its neighborhood vector $R(v_i) = \{c_{i1}, \ldots, c_{it}, c_{1i}, \ldots, c_{ti}\}$. For RRE, t is the number of roles out of the upper-level SA partition. For SSE, t is the number of nodes in the network. For unweighted RRE/SSE, $c_{ij}(c_{ji}) = \{0, 1\}$ denotes whether the jth role/node is present in the outbound (inbound) neighborhood set of node v_i. For weighted RRE, $c_{ij}(c_{ji})$ denotes the number of occurrence of the jth role in the neighborhood of node v_i. For weighted SSE, $c_{ij}(c_{ji})$ denotes the weight of the edge connecting v_i and v_j. In a normalized setting, the neighborhood vector is refined by $c_{ij} = c_{ij}/\sum_{j=1,\ldots,t}(c_{ij} + c_{ji})$.

Next, over all the nodes to be partitioned, a pairwise similarity score is computed using the node's neighborhood vector. Though there are many candidate criteria, we choose the Euclidean distance, because we care the similarity in both orientation and magnitude. Finally, to compute the fuzzy equivalence partition, we apply the k-means clustering algorithm (MacQueen, 1967).

4.5. Implementation and performance

For deterministic OnionGraph abstractions, we introduce a unified method to compute the partition at all five levels. The core concept is the design of the row vector, representing both the semantic and topological information on each node. As shown in Fig. 5, the row vector is composed of the node attribute (type) field, the neighborhood relationship, as well as an explicit node identifier when the network is partitioned into per-node groups. The extensions of partition algorithms, e.g., directed and weighted partitions, are supported by design. Finally, the network partition is achieved through an appropriate hash function over the row vectors of all the nodes. This implementation has a linear complexity of $O(m + dn)$, where n, m, d are the number of nodes, links, and node attribute values of the input network.

The fuzzy equivalence computation using the k-means clustering has an intrinsic complexity of $O(k \cdot n \cdot d \cdot l)$. Here k is the number of desired clusters and l is the number of iterations in the computation, generally small for most graphs. n is the number of nodes in the network. d is the maximal number of dimensions of the neighborhood vector. For fuzzy RRE, k, l and d are bounded, therefore still holds a linear computational complexity. For fuzzy SSE, $d = n$, the complexity is quadratic to the number of nodes and is slow for large networks.

Table 1
OnionGraph network abstraction performance.

Dataset	Version	#Node	#Link	SA (s)	RRE (s)	SSE (s)
VASTC	D	2596	36669	0.058	0.19	0.25
Vis-Bibli.	D	20615	106316	0.22	0.75	1.16
Vis-Bibli.	F	20615	106316	N/A	1.04	111.89
Twitter	D	306126	1424427	2.26	4.60	12.20
Twitter	F	306126	1424427	N/A	7.08	>1000
Honeypot	D	1051595	1158150	12.06	19.90	27.43
Honeypot	F	1051595	1158150	N/A	59.08	>1000

We evaluate the OnionGraph performance on a Windows desktop (Quad-core Intel Xeon@3.30 GHz with 6 GB of memory). Four heterogeneous network data sets from medium to large size are used as the input for the abstraction, as shown in Table 1. In the same table, the abstraction time by SA, RRE and SSE partitions are presented. These results suggest that our theoretical analyses correspond well with the actual performance. The completion time of the deterministic version (D) of all three partitions is almost linear to the number of nodes and links. The slowest SSE partition completes in 27 s on a network with a million nodes and links. In the fuzzy version (F, with five clusters) of partitions, there are moderate penalties on the RRE, but the running time becomes too long for SSE, even on a medium-size 20 000-node network.

5. Information-theoretic navigation

In the initial proposal, the OnionGraph navigation is achieved through the focus+context user interactions. These interactions can provide a full flexibility to customize the network abstraction based on the analysis tasks, however, require user's prior knowledge on the underlying network to plan the navigation path. This can be difficult, if not impossible, for users working with either a new data set or explorative analysis tasks.

Motivated by this problem, we develop another suite of navigation techniques that automatically recommend the optimal navigation path and guide users in their network discovery trail. The key idea is to model the multivariate network navigation process using the information theory. Recently, information theory has been introduced to the visualization community to interpret the visualization process (Chen and Jäenicke, 2010) and to correspond elements of data visualization with those in the data communications (Wang and Shen, 2011). Specially in scientific visualization, the concept of entropy is used to locate important regions in the volumetric data, such as selecting the optimal view (Bordoloi and Shen, 2005; Takahashi et al., 2005), improving the LOD map (Wang and Shen, 2006) and automatically focusing on important features (Viola et al., 2006). Similarly in the information visualization, information theory has been applied to help present and explore multivariate and time-varying data sets (Biswas et al., 2013; Wang et al., 2008, 2011). Despite the existing literature, to our knowledge, the principle of information theory has not been considered in the navigation of multivariate networks.

5.1. Information entropy model

Denote the original multivariate network by a random variable G and its abstraction under a given OnionGraph setting by another random variable C. Without loss of generality, all instances of G are assumed to be undirected and unweighted. The directed case will be a plain extension of the study here, while

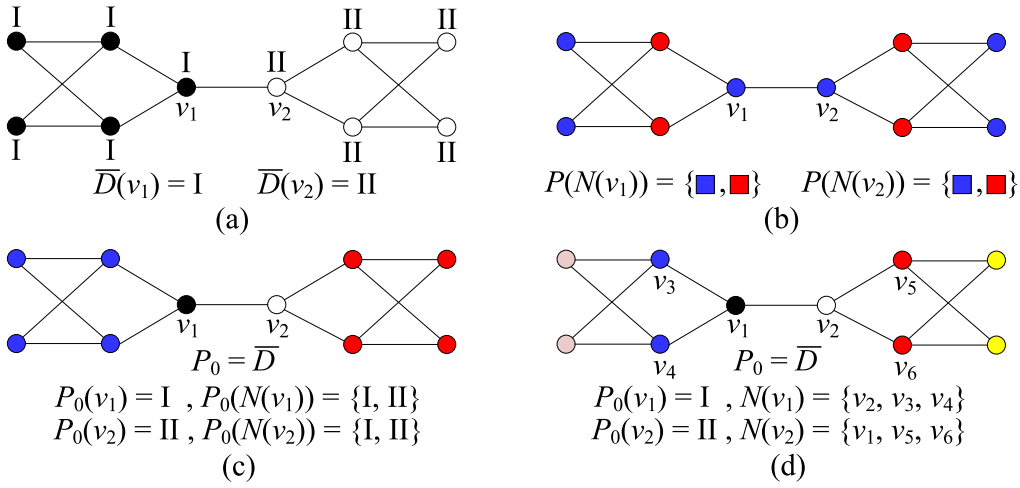


Fig. 4. OnionGraph partitions on the undirected network. In each subfigure, node colors indicate the partition index: (a) semantic aggregation, the selected attribute value is labeled on each node; (b) a sample of the regular equivalence partition; (c) the regular equivalence partition relative to the semantic aggregation in (a); (d) strong structural equivalence partition. (For interpretation of the references to color in this figure legend, the reader is referred to the web version of this article.)

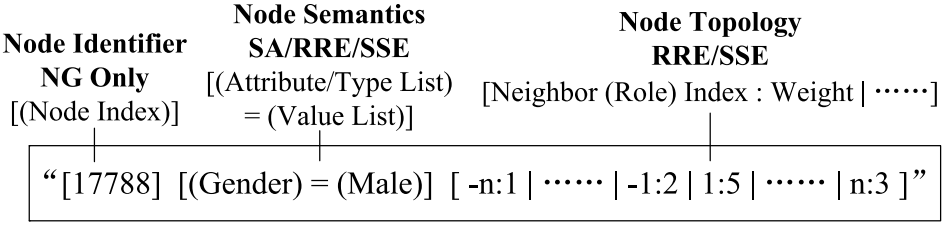


Fig. 5. Row vector design in OnionGraph. NG = per-node group.

the weighted case is more sophisticated and will be a subject of future work. We derive the information gain of seeing G and C by their information entropies which are defined as $H(G)$ and $H(C)$. Take $H(G)$ as an example, by the information theory, it is written as:

$$H(G) = -\sum_i p_i \log p_i \tag{4}$$

where p_i denotes the probability for G to be its ith feasible graph instance g_i.

The computation of $H(G)$ and $H(C)$ can be complicated, which relates to the theory of graph enumeration and compression. Fortunately, to model the navigation process of OnionGraph, we do not need to calculate $H(G)$ and $H(C)$. Instead, we focus on the conditional entropy of $H(G|C)$, which defines the remaining information of G given (i.e., after seeing) the instance of the abstraction C. By the chain rule of information theory (Cover and Thomas, 2006), we have

$$H(G, C) = H(C) + H(G|C) = H(G) + H(C|G) \tag{5}$$

where $H(G, C)$ represents the joint entropy of G and C. Because the OnionGraph abstraction C is uniquely constructed from the original network G if the OnionGraph setting is known, we have $H(C|G) = 0$. Eq. (5) becomes

$$H(G|C) = H(G) - H(C) \tag{6}$$

This shows that $H(G|C)$ explicitly quantifies the additional information users can obtain from G beyond the initial abstraction C. The conditional entropy also captures the maximal possible information gain during the navigation process over C. On the opposite side, $H(G|C)$ can be viewed as the uncertainty of the graph abstraction C given the many possible original full network G. Fig. 6 illustrates the relationship of the above mentioned quantities, as well as their dynamics during the OnionGraph navigation process. When the initial abstraction of C is expanded gradually to the full network G, users gain more information whereas the uncertainty of the abstracted visualization drops.

In the application, the value of $H(G|C)$ is not enough to serve as the guide to the network navigation process, because users need to know where to expand/collapse on the abstraction. We follow up to decompose $H(G|C)$ into $\sum_i H(\pi_i|C)$, where $\pi_i \in C$ denotes the ith node group in the abstraction C and $H(\pi_i|C)$ denotes the conditional entropy of the connections of all the nodes in π_i given C. $H(\pi_i|C)$ represents the uncertainty attached to the node group π_i. Computing $H(\pi_i|C)$ involves the examination of all possible connection cases of nodes in π_i. Consider the network abstraction in Fig. 7(a) as an example, the node group π_0 is expanded into three original nodes $\{v_1, v_2, v_3\}$. A more general case is given in Fig. 7(b) where the node group π_i contains N original nodes $\{v_1, \ldots, v_N\}$ and connects to d adjacent node groups $\pi_1, \ldots, \pi_d$. The d corresponding edge groups have the size of $M_1, \ldots, M_d$. The loop edge group from π_i to itself has a size of M. By the principle of maximum entropy, we assume an equal probability for each connection case, and compute $H(\pi_i|C)$ by

$$H(\pi_i|C) = -\sum_i p_i \log p_i = -\sum_{i=1}^{\#Case} \frac{\log \frac{1}{\#Case}}{\#Case} = \log \#Case \tag{7}$$

Here $\#Case$ denotes the number of possible connection cases of π_i. To compute $\#Case$, we notice that in the general setting of Fig. 7(b), each possible connection case of π_i can be defined by $N + 1$ case variables. The first N variables are the neighborhood vector of all original nodes in π_i, denoted by R_j for v_j ($j \in [1, N]$). R_j defines the number of connections between v_j and all the d neighboring node groups of π_i. The last case variable by $G(\pi_i)$

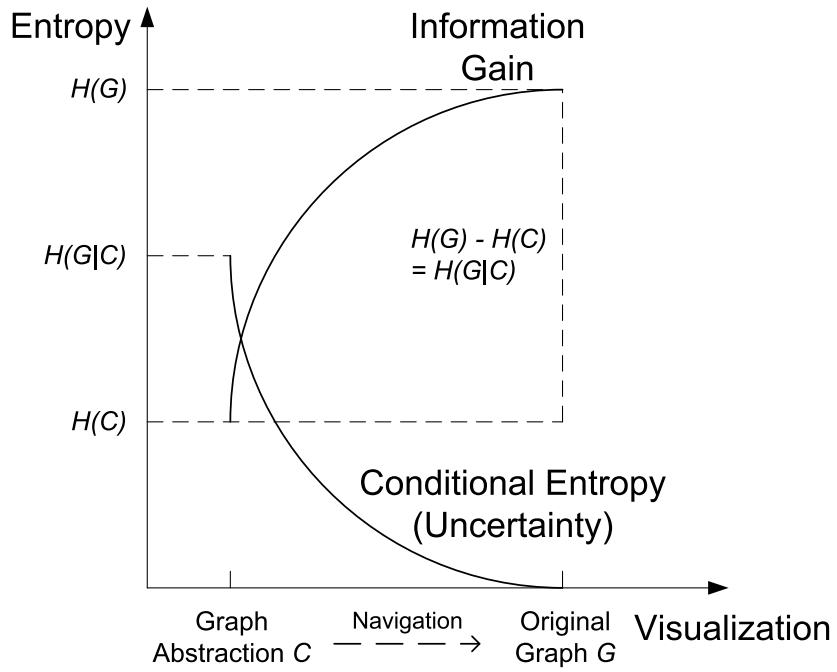


Fig. 6. The dynamics of information gain and uncertainty in the navigation from the abstraction (C) to the original network (G).

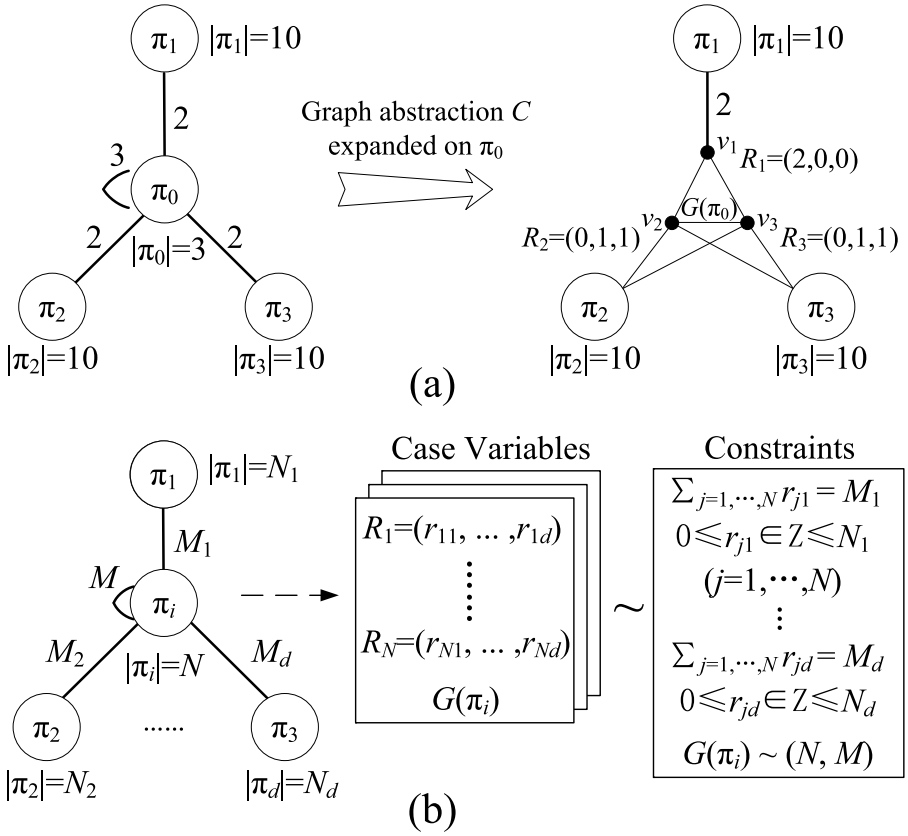


Fig. 7. The expansion of node groups in the OnionGraph abstraction. The size of node/edge groups are labeled unless a trivial size of one: (a) a sample case; (b) the general case.

defines the internal graph structure of π_i. Each connection case is feasible if its case variables satisfy a few constraints subject to the given size of node groups $(N_1, \ldots, N_d)$ and edge groups $(M_1, \ldots, M_d)$. As shown in Fig. 7(b), the constraints by the group edge from π_i to π_1 is exactly a restricted version of the weak composition of M_1 by N nonnegative integers (Eger, 2013). In Appendix A, we show that the restriction by N_1 will not reduce the number of feasible cases much. The computation of *#Case* then degenerates to d standard weak composition problems. The number of cases by solving each problem is given by (8). Additionally,

on the internal graph structure of π_i, the expanded graph should have exactly N nodes and M edges, where the number of cases is computed by (9). Approximations are introduced assuming a large N.

$$\#Case\_by\_(M_j, N_j) = \binom{M_j + N - 1}{N - 1} \approx \frac{(M_j + N)!}{N! M_j!} \tag{8}$$

$$\#Case\_by\_G(\pi_i) = \binom{N(N-1)/2}{M} \approx \frac{(N^2/2)^M}{M!} \tag{9}$$

Summing together, the conditional entropy of π_i becomes

$$\begin{aligned} H(\pi_i|C) &= \log(\#Case\_by\_G(\pi_i) \times \prod_{j=1,\ldots,d} \#Case\_by\_(M_j, N_j)) \\ &= M \log N^2/2 - \log M! N!^d + \sum_{j=1,\ldots,d} [\log (M_j + N)!/M_j!] \end{aligned} \tag{10}$$

The computational complexity is $O(M + N + \max M_j)$. Using Stirling's approximation in (11), it is reduced to $O(1)$.

$$\log N! \approx N(\log N - 1) + \log \sqrt{2\pi N} \tag{11}$$

In the OnionGraph visualization, $H(\pi_i|C)$ is displayed on each node group as a visual hint of the unobserved information beyond the network abstraction C. As shown in Fig. 9(b), the author and paper node group with larger conditional entropies will be drawn in more opaque colors than all the venue nodes, which contain zero unobserved information. On the other hand, during the interactive analysis with OnionGraph, users are more likely to proceed from an initial abstraction to an expansion state in the middle. They hardly reach the full details of the original network. In this sense, $H(\pi_i|C)$ is still not the most appropriate metric to guide each single step of the network exploration. In the following, we base on the proposed information entropy model and introduce two new information-theoretic OnionGraph navigation methods by maximizing the change of the conditional entropy. These methods provide users with the largest information gain in their navigation process.

5.2. Guided semantic aggregations

On the OnionGraph abstraction C, consider the scenario that users want to drill down on the node group π_i to check its detailed connections and suppose that they will apply the semantic aggregation in the next-level abstraction. In case the node attribute D is used, we define the corresponding sub-groups expanded from π_i by $W = \{\omega_1, \ldots, \omega_c\}$. User's information gain in this process can be modeled by the decrease of the conditional entropy:

$$I(\pi_i \to W|C) = H(\pi_i|C) - \sum_{i=1}^{c} H(\omega_i|(C - \pi_i) \cup W) \tag{12}$$

where $(C - \pi_i) \cup W$ denotes the network abstraction after expanding π_i to W. $H(\omega_i|(C - \pi_i) \cup W)$ can be computed similar to $H(\pi_i|C)$.

On exploring the abstraction C with the semantic aggregation, the key user choice is which node attribute to apply in the expansion. Denote all feasible node attributes by $D_1, \ldots, D_d$ and the corresponding sub-groups expanded from π_i by $W_1, \ldots, W_d$. According to the information theory, the attribute that leads to the largest information gain will be the best choice. This is then recommended to the user. The optimal node attribute can be formally defined as

$$\underset{j}{argmax}\ I(\pi_i \to W_j|C) \tag{13}$$

In the visualization design, after users select part of the network abstraction for the semantic aggregation, the information gain by applying each feasible node attribute is pre-computed and visually mapped to the attribute selector as the information hint. An example is shown in the OnionGraph abstraction control panel of Fig. 9(b).

5.3. Optimal network partitions

In another scenario, users choose to adopt the fuzzy equivalence based partition (the level-III/IV OnionGraph abstraction) on the node group π_i. They need to specify the number of sub-groups to divide, which can be difficult without the prior knowledge on the underlying network. Here we apply the information-theoretic approach, which automatically computes the optimal number of sub-groups by maximizing the information gain in the navigation process. Note that there is a subtle difference from the case of the semantic aggregation. As the number of sub-groups increases, the perceived information gain will grow asymptotically to the full value of $H(\pi_i|C)$. The maximal gain will be achieved when the node group π_i is split thoroughly into original nodes, which may not meet the user's requirement.

To solve this problem, we introduce a new information-theoretic metric, namely the information efficiency, which is defined as the average information gain obtained by each new sub-group (visual element). Suppose the node group π_i is divided into k sub-groups defined as $W^{(k)}$, the final choice to maximize the information efficiency is given by

$$\underset{k}{argmax}\ \frac{I(\pi_i \to W^{(k)}|C)}{k - 1} \tag{14}$$

6. Visualization

Fig. 1 illustrates the OnionGraph user interface. It is composed of three parts: OnionGraph visualization in the center (Section 6.1), the control/filter panel on the left and the legend/list/detail panel on the right (Section 6.2).

6.1. Oniongraph visual metaphor

A typical OnionGraph visualization is shown in Fig. 8 by abstracting the bibliographic network. Each node in the view represents a group of individual nodes from the original network. The initial SA abstraction aggregates all the nodes into three type-based groups ("author", "paper" and "venue", as shown in Fig. 9(a)). These groups are drawn in filled nodes where the fill color and the icon on the top-right of each node indicate the node type. In Fig. 8, the spring-green node in the center represents all 9557 papers. All the other nodes in Fig. 8 have been drilled down from the top-level heterogeneous abstraction. They are drawn by the "onion" metaphor composed of several concentric circles. The number of circles indicates the abstraction hierarchy: the SA group on node attributes has three circles (e.g., the venue nodes in Fig. 8), RRE has two (e.g., the author nodes in Fig. 8), SSE has one, the individual node only leaves a solid dot. Upon the top-down exploration, the visual complexity of each node group in OnionGraph is reduced whereas the number of node groups increases, so as to keep the overall complexity of the OnionGraph view in a sustainable level.

In OnionGraph visualization, the size of the node encodes the number of individual nodes inside the group. Note that we apply normalization here, each group size is divided by the total number of nodes in the same node type. The visual result is a balanced view that will not bypass the minority node type (e.g., the venues) and still show variations on the group size. The color of each node

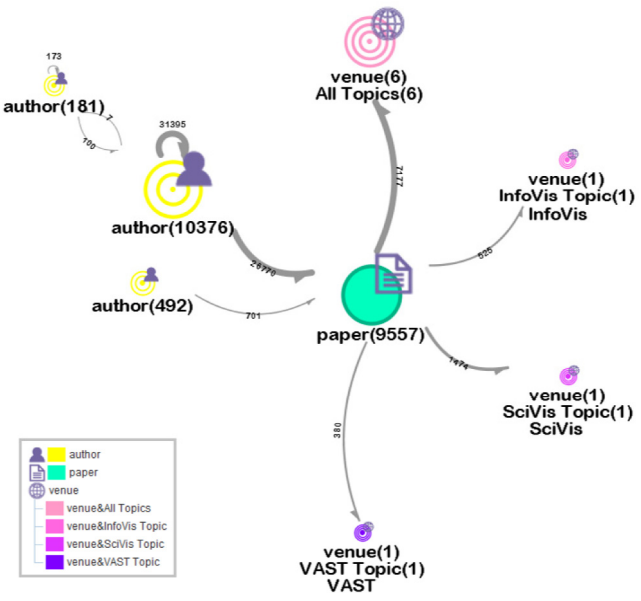


Fig. 8. An OnionGraph visualization of the author-paper-venue bibliographic network in the visualization community. Three yellow groups indicate the authors with different connection patterns: normal authors with co-authors and publications, special authors who only write single-authored papers, and anomalous authors without a publication (potential errors in the data set). Four indigo groups indicate the venues (conferences/journals) on different topics. The spring-green group indicates all the papers. (For interpretation of the references to color in this figure legend, the reader is referred to the web version of this article.)

is determined by the type/attribute values of each node group. Initially on the AS abstraction by node type, the colors are picked uniformly on the color hue. After the expansion of the top-level node group into sub-groups, new colors are assigned with linear hue and saturation offsets from the original group. An example is shown by the legend in the bottom-left part of Fig. 8. The node label by default displays the value of the node type/attributes used for the abstraction. When the node group contains only one node, the node title is used as the label. The selected nodes are drawn with dark-red outlines and labels, coupled with a "+/−" sign upon mouse hovering to indicate the potential lower/upper level to explore. The link thickness and the link label encode the number of individual links between the node groups. Different from the ordinary network, OnionGraph usually has a loopback link on each node group indicating the internal connection, as shown by the arc above the node.

6.2. Interaction

Apart from the traditional interaction methods on the network visualization (node selection, drag&drop, etc.), the OnionGraph interface provides users two extra controllers to interact with the OnionGraph visualization. First, the abstraction control panel as shown in Fig. 1(b). After users select the interested part of the network, they can specify a new abstraction profile in the control panel, including setting the abstraction level, selecting the node attributes and turning on/off several switches of the abstraction profile (e.g., directed, weighted and fuzzy versions of RRE/SSE). The selected network is processed after clicking the "abstract" button, and finally shown as the finer/coarser-grained visual abstraction. In another usage, users can double-click on the selected nodes to expand/collapse to the lower/upper level of abstraction. Before the abstraction, users can plug in node/edge attribute filters as in Fig. 1(c). These filters work in a global manner and generate the input for the OnionGraph abstraction.

Second, through the pop-up menu of the OnionGraph interface, visual parameters can be configured, such as the layout algorithm, the node/link visual encodings. Notably, OnionGraph allows a neighborhood charting mode. As shown in the left column of Fig. 1(a), each node group abstracted by RRE is drawn by a chart instead of the onion metaphor. These charts illustrate the distribution of attribute values in the node's neighborhood. The right part of the OnionGraph interface shows network details upon the visualization and user interaction. The top-right panel (Fig. 1(d)) displays the node legend indicating the icon/color assigned to each node group. The center-right panel (Fig. 1(e)) displays the list of nodes currently selected in the main view. Upon choosing one node in the list, the node attributes are displayed in a key–value table in the bottom-right panel (Fig. 1(f)).

6.3. Network layout

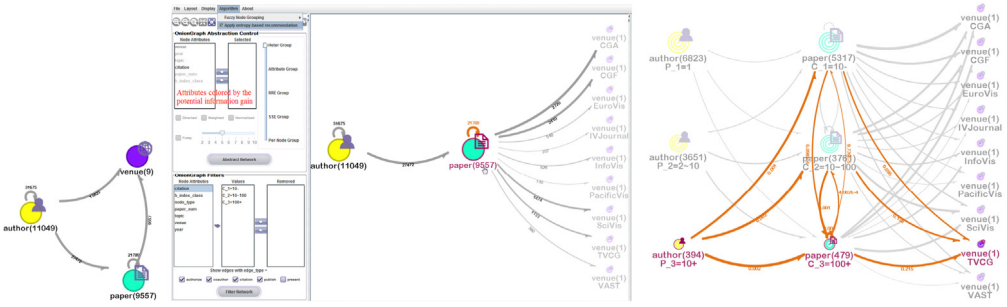
We design two kinds of layouts for OnionGraph. The first is a grid-based layout improved from PivotGraph (Wattenberg, 2006). The initial PivotGraph layout explicitly selects two node attributes and places each node by its attribute values. On each OnionGraph view, there can be more than one abstraction profiles, each managing part of the network. Applying two global node attributes may not meet the nature of all abstraction profiles. In our improved algorithm, we pick only one global attribute, i.e., the node type, which is mapped to the X axis of the layout. After that, each group of nodes with the same abstraction profile selects their own second node attribute, which is mapped to the Y axis. An example result is shown in Fig. 1(b). This grid-based layout works for most OnionGraph settings, but does not guarantee an efficient use of the space. Therefore, we also implement the force-directed layout which optimizes the space utilization and highlights the network topology.

7. Case study

7.1. Academic network

In the first case study, we apply OnionGraph to analyze the academic network of the visualization community. The data set was extracted from ArnetMiner (Tang et al., 2008), which includes scientific papers at nine major visualization conferences and journals (SciVis, InfoVis, VAST, TVCG, etc.). Each paper entry in the database has multiple attributes: title, authors, publication venue, date, citations, keywords, abstract, etc. We built a multivariate bibliographic network with three node types: 11 049 authors, 9557 papers and nine venues. Five types of links are identified: the co-authorship among authors, the citation between papers, the author-paper affiliation, the publication of a paper in a venue, and the presentation of an author in a venue. Apart from the existing data fields, more node attributes are derived by analytics: the papers are classified into 10 topics using LDA (Blei et al., 2003) on their textual content; each author is computed an h-index from his paper citations in the community.

We invited a senior visualization researcher to use the OnionGraph tool to explore the academic network and gain insights. Initially, he was provided with the default overview in the heterogeneous abstraction level, as shown in Fig. 9(a). He proceeded to expand the venue group by venue name in the SA abstraction level and obtained Fig. 9(b). The layout was changed to the grid-based one for clarity. The venues with the highest number of papers are CG&A and CGF (the right column of Fig. 9(b)), which both publish more than 2000 papers as shown by the edge labels. The node/edge lightness indicates the uncertainty on the



(a) Initial view (b) Venue group expanded (c) Paper/author expanded by citation/paper category

Fig. 9. Academic network analysis in the visualization community.

node/edge group. The author and paper group have not been expanded and are drawn in the most opaque color to represent the largest uncertainty. Meanwhile, the nine venue nodes are fully expanded and there is no more uncertainty, so their node colors are in the lowest lightness scale. To continue the analysis, he needed to decide where and how to explore next. After selecting the paper node group, OnionGraph pre-computed all the feasible node attributes for the next SA abstraction. The information gain by applying each attribute is visually mapped to the selection panel, as shown in the top-left list of Fig. 9(b). Following the recommendation of the system, he expanded the paper node into three sub-groups by their citation counts: low (<10), medium ($10 \sim 100$) and high-citation (>100). In the same way, the author group was expanded by the number of paper published. The OnionGraph view becomes Fig. 9(c). He had switched the mapping of link thickness to the average number of links in a link group, aka the probability of each paper published in a venue. This measure is more relevant to the academic performance. From Fig. 9(c), he found through interactions that though CG&A and CGF published a lot of medium and high-citation papers, their shares in these two groups (CG&A: 0.22, 0.18; CGF: 0.24, 0.21) are lower than those in the low-citation papers (CG&A: 0.34; CGF: 0.26). In comparison, TVCG (highlighted node in the right column of Fig. 9(c)) has a dramatic increase in the share of medium and high-citation papers (0.14, 0.22) than the low-citation papers (0.1). In the following, he applied a filter to get rid of the author groups with low publications ($\leq$10 paper) and located 394 most active people in the visualization community. He further analyzed their citation performance by a lower-level fuzzy RRE abstraction and the entropy-based navigation. These active authors were automatically classified into four sub-groups according to their publication numbers in different citation groups, as shown in Fig. 1(a). The onion metaphor on the author nodes was switched to the neighborhood charts to illustrate their distribution patterns. The sub-group with the largest author icon indicates 309 active authors whose average publication includes a few low and medium-citation papers (7.4, 6.9) and almost only one high-citation paper. The second largest group contains 75 authors with more low, medium and high-citation papers (17.7, 16.5, 2.5 on average). The next group (focused node in Fig. 1(a)) probably indicates seven long-standing fellows in the community, who published 40.7 low, 47.6 medium and 5 high-citation papers on average. Interestingly, there is another small group (3 authors) who published 74 papers on average, but only 0.3 of them are high-citation papers.

7.2. Brain network

In another case, we study the human brain network created by multimodal Magnetic Resonance Imaging (MRI) (Gray et al., 2012), also known as the human connectome. The data set contains the brain network of 113 people, each consisting of 70 cortical regions as nodes and the fiber connections between these regions as links. Each link is measured with a fiber strength, and the brain network becomes a undirected, weighted graph. The multivariate nature of the network comes from the demographics of each people, including age, gender, the intelligence level by full-scale IQ (FSIQ), and the degree of personality traits (Openness, Extraversion, etc.). We classify these numerical measures into categories, e.g., the raw FSIQ is mapped into four intellectual grades.

An example of the 70-region connectome is shown in Fig. 10(a). Over all the 113 brain networks, the number of connections in each network ranges from 800 to 1200, forming very dense graphs. To adapt to this nature, we apply the OnionGraph tool with the attribute-based abstraction, edge statistics visualization, but do not use the topology analysis by SSE/RRE.

Consider an investigator attempting to analyze the correlation between people's connectome and their demographics such as FSIQ. He starts from aggregating all the 113 brain networks by the node region index, as shown in Fig. 10(b). This abstraction is similar to the single brain network, except for the higher node degree and the larger link density. Mapping the number of original link into the edge color lightness reveals an overall pattern that quite a lot connections are shared by most people, as shown in Fig. 10(c). Almost a half of links are shared by at least 50 people in the data set. The investigator further compares brain networks of different FSIQ categories by the expansion operation. The comparative graphs with edge color lightness showing the average fiber strength is illustrated in Fig. 11. Initially, it is clear that the connectome of different FSIQ scales are almost the same, even considering the fiber strength. A further analysis with edge statistics filter discovers more interesting findings. The investigator first applies filters to only leave the popular connections shared by more than a half of people in each group, and again he notices no significant difference among the different FSIQ groups. However, when he reverts the filter to only leave the bottom 25% connections that are less frequently shared by individuals, clear patterns relating to the ordinal characteristics of the FSIQ group are identified. As shown in Fig. 12, the lower the FSIQ (talented$\rightarrow$super$\rightarrow$high$\rightarrow$average), the stronger fiber connections that are shared by only a minority group of people. This may indicate the negative impact of minority fiber connections in the brain network to the human intelligence.

7.3. Security network

The security Host-User-Application (HUA) network is generated in a typical lab setting. There are four basic node types: *H* node denotes the host, which is further partitioned into internal

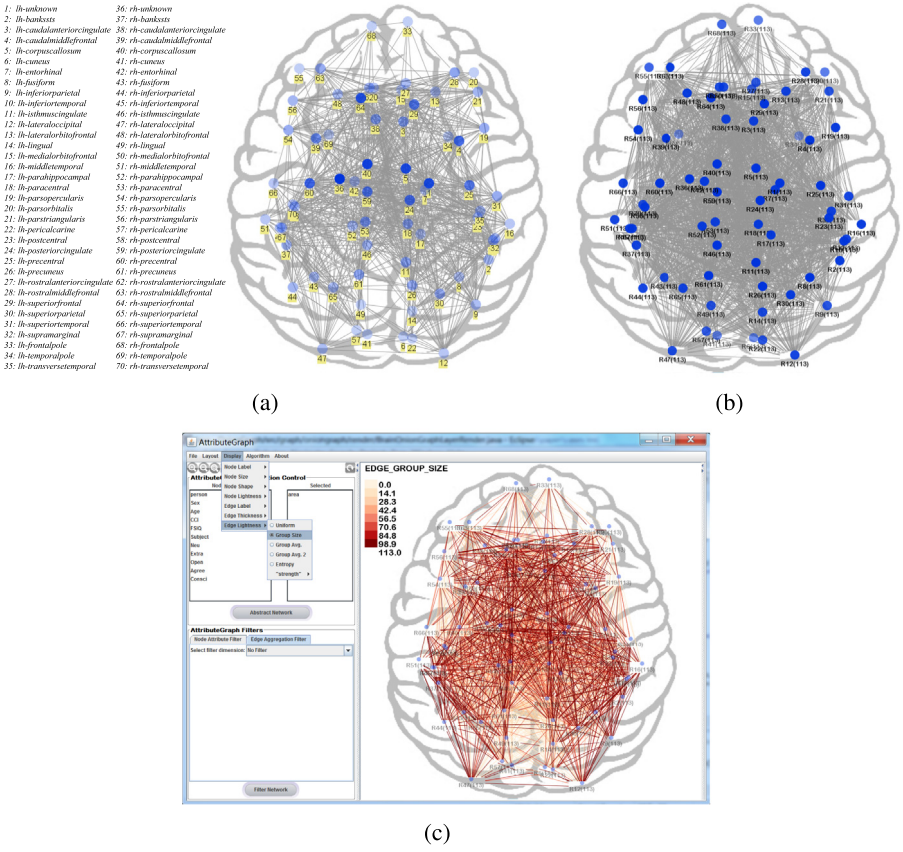


Fig. 10. Brain network visualizations with nodes representing cortical regions and links indicating fiber connections. The node layout is using the central of each brain region in the top-down view, i.e. (X, Y) out of the 3D coordinate: (a) The connectome of one people, the node color shows the node degree in the graph and the node label indicates its brain region index (a full region list is given in the left); (b) The aggregated connectome of 113 people in our data set, nodes are grouped by the region index; (c) Visually map the number of original link in each link aggregation the color lightness scale.

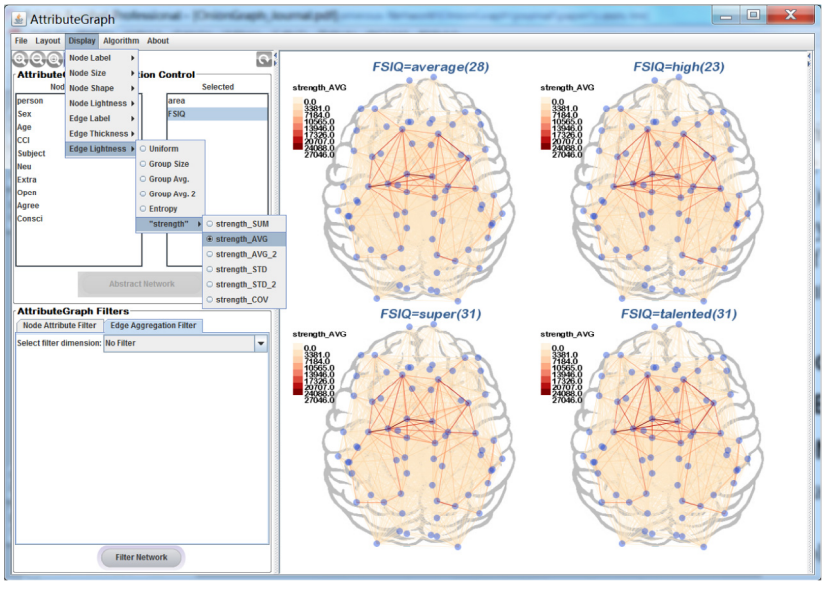


Fig. 11. The comparative brain network view of different FSIQ categories. Link color indicates the average fiber strength. The numbers in parentheses (e.g., average(28)) indicate the number of subjects in each group.

hosts in the Intranet and external domains in the Internet. *U* node denotes the user connectivity (usr). *A* node denotes the application connectivity (app).

We recruited a network administrator to analyze his own lab traces with OnionGraph. He started with the typical HUA network in Fig. 13(a). From the graph, he found that there were 128 users logged on 601 internal hosts running 298 unique applications, which connected either internal hosts or 2802 external domains. He had a few interesting observations when moving from the initial heterogeneous abstraction to the RRE abstraction on each

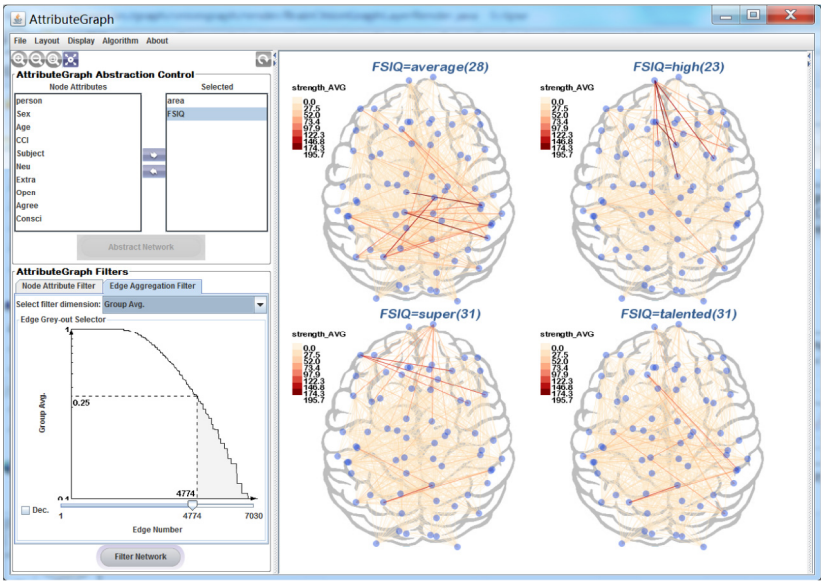
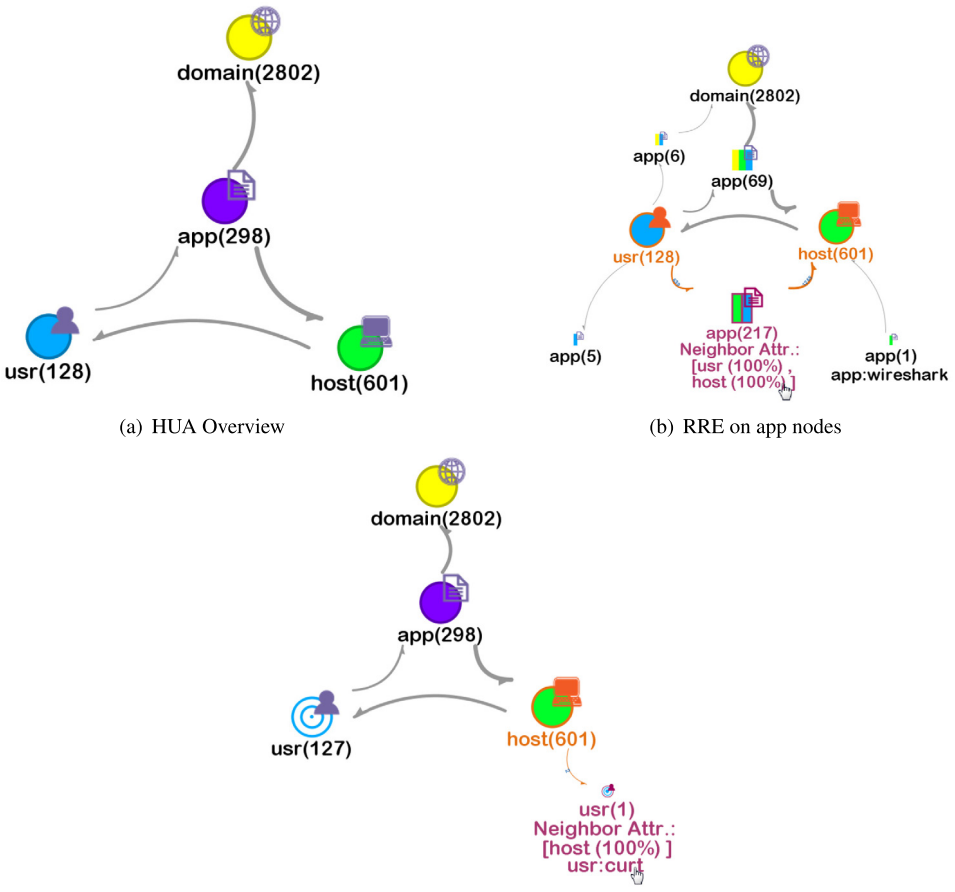


Fig. 12. The filtered view that only compares the bottom connections shared by less than 25% people in each FSIQ group.



(a) HUA Overview

(b) RRE on app nodes

(c) RRE on user nodes

Fig. 13. HUA communication network visual analytics.

node type. First, the app nodes were split into five sub-groups, as shown in Fig. 13(b) displayed by the neighborhood charts: (1) the majority of apps (217) connected to only internal hosts by users (focused node in the graph); (2) 6 apps connected to only external domains by users; (3) 69 apps connected to both internal hosts and external domains by users; (4) 5 apps did not make network connections; and (5) one app run by an unknown user talked to a few internal hosts. Type-1 apps contain predominantly scientific computing programs while Type-2 and Type-3 apps have significantly more generic network applications such as ssh, firefox, ftp, etc. In particular, the Type-5 node containing only one app (wireshark) is clearly suspicious, possibly leveraged by a

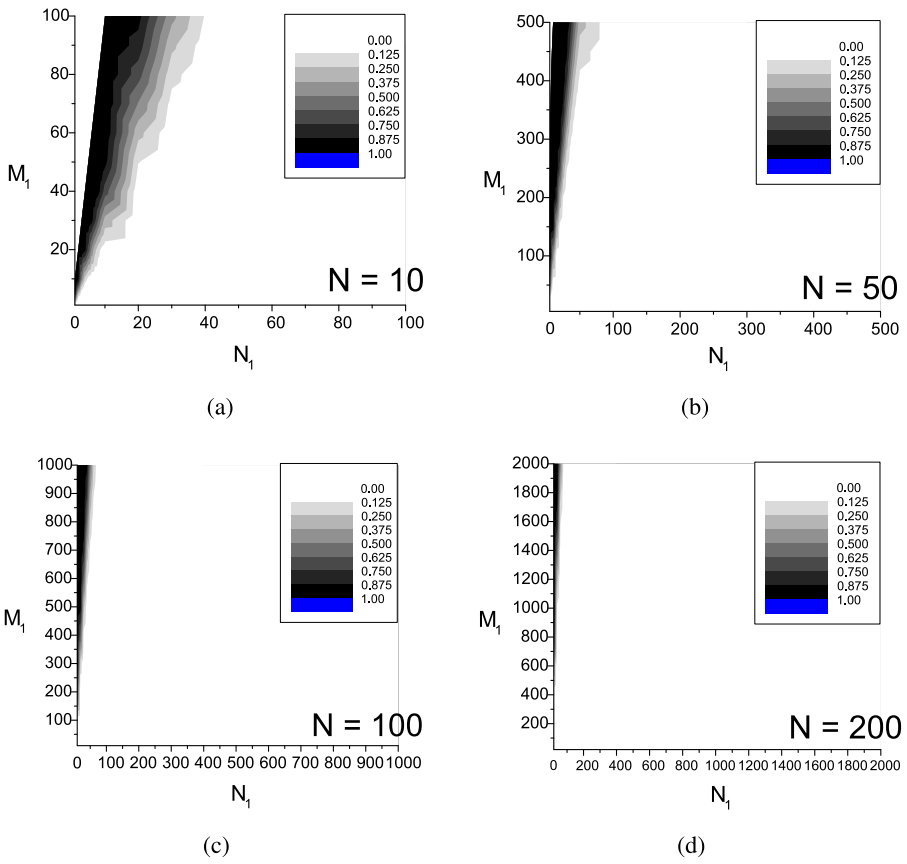


Fig. A.14. The approximation error rate using the unrestricted weak composition. Darker color indicates a larger error rate.

malicious user to sniff packets on the network. Second, the user node had been divided into two groups (Fig. 13(c)): (1) 127 users that had run apps to connect to other computers; and (2) the only user who never ran apps. The Type-1 users are primarily enterprise users who are allowed to run scientific programs. The Type-2 user is the system administrator. It is clear that normal users and privileged users have distinguished activity patterns.

8. Conclusion

OnionGraph is a visual analysis framework for the exploration of heterogeneous multivariate networks. It is realized by scalable algorithms creating attribute-based and various structural equivalence network partitions. By combining semantic and topological information for a hierarchical abstraction, OnionGraph enables the level-of-detail viewing of large multivariate networks. The navigation and filtering interactions in complement to each other are shown to be effective in customizing the OnionGraph analysis process. The evaluation result in case studies demonstrates that OnionGraph is useful in many multivariate network analysis scenarios where the task is exploratory and involves both attribute-centric and structural problem solving.

Declaration of competing interest

The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

Acknowledgments

This work was supported by NSFC Grants 61772504 and the Fundamental Research Funds for the Central Universities. Weihong Qian is the corresponding author.

Appendix A. Approximation of the S-restricted weak composition problem

After a full expansion of the group node π_i in the graph abstraction C, each possible connection case corresponds to multiple S-restricted weak compositions of number $M_j (j = 1, \ldots, d)$ and a sample graph out of π_i. We focus on the composition of M_1 as an example, which is defined by (A.1). The restriction in (A.2) describes the constraint by the size of the group node in the other endpoint.

$$\sum_{j=1,\ldots,N} r_{j1} = M_1 \tag{A.1}$$

$$r_{j1} \in S = \{0, 1, \ldots, N_1\}, \forall j = 1, \ldots, N \tag{A.2}$$

Then the number of cases is given by the extended polynomial coefficient:

$$[x^{M_1}]\Big(\sum_{a=0,1,\ldots,N_1} x^a\Big)^N$$

According to Eger (2013), when the restriction is part of $\{0, 1, \ldots, N_1\}$, the extended polynomial coefficient can be computed by

$$\#Case\_by\_(M_1, N_1)\_restricted = \sum_{j=0,1,\ldots,N} (-1)^j \binom{N}{j} \binom{M_1 + N - (N_1 + 1)j - 1}{N - 1} \tag{A.3}$$

In the worst case, this needs $O(N^2)$ time to calculate. We then look at the unrestricted version, motivated by the assumption that $r_{j1} > N_1$ happens in rare cases. The unrestricted weak

composition leads to the number of cases

$$\#Case\_by\_(M_1, N_1)\_unrestricted = \binom{M_1 + N - 1}{N - 1} \tag{A.4}$$

To validate our assumption, we conducted numerical simulations to compute the error rate (denoted by ρ) of using unrestricted calculation to approximate the restricted number:

$$\rho = \frac{\#Case\_by\_(M_1, N_1)\_unrestricted - \#Case\_by\_(M_1, N_1)\_restricted}{\#Case\_by\_(M_1, N_1)\_restricted} \tag{A.5}$$

Results in Fig. A.14 show that the approximation only leads to a significant error when N (the size of the node group to expand) is small and M_1 (the size of the connecting group edge) is much larger than N_1 (the size of the node group in the other endpoint). By looking at the data set, we confirm that such cases happen extremely rarely.

Appendix B. Supplementary data

Supplementary material related to this article can be found online at https://doi.org/10.1016/j.visinf.2020.01.002.

References

Abello, J., van Ham, F., Krishnan, N., 2006. ASK-Graphview: A large scale graph visualization system. IEEE Trans. Vis. Comput. Graphics 12, 669–676.

Archambault, D., Munzner, T., Auber, D., 2008. Grouseflocks: Steerable exploration of graph hierarchy space. IEEE Trans. Vis. Comput. Graphics 14, 900–913.

Auber, D., 2004. Tulipła huge graph visualization framework. In: Graph Drawing Software. Springer, pp. 105–126.

Auber, D., Chiricota, Y., Jourdan, F., Melancon, G., 2003. Multiscale visualization of small world networks, In: InfoVis'03, pp. 75–81.

Battista, G.D., Eades, P., Tamassia, R., Tollis, I.G., 1998. Graph Drawing: Algorithms for the Visualization of Graphs. Prentice Hall PTR, Upper Saddle River, NJ, USA.

Bezerianos, A., Chevalier, F., Dragicevic, P., Elmqvist, N., Fekete, J.D., 2010. Graphdice: A system for exploring multivariate social networks. Comput. Graph. Forum 29, 863–872.

Biswas, A., Dutta, S., Shen, H.W., Woodring, J., 2013. An information-aware framework for exploring multivariate data sets. IEEE Trans. Vis. Comput. Graphics 19, 2683–2692.

Blei, D.M., Ng, A.Y., Jordan, M.I., 2003. Latent dirichlet allocation. J. Mach. Learn. Res. 3, 993–1022.

Bordoloi, U.D., Shen, H.W., 2005. View selection for volume rendering. In: IEEE Visualization, pp. 487–494.

Cao, N., Sun, J., Lin, Y.R., Gotz, D., Liu, S., Qu, H., 2010. Facetatlas: Multifaceted visualization for rich text corpora. IEEE Trans. Vis. Comput. Graphics 16, 1172–1181.

Card, S.K., Mackinlay, J., Shneiderman, B., 1999. Readings in Information Visualization: Using Vision to Think, first ed. Morgan Kaufmann, ISBN: 1558605339.

Chen, M., Jäenicke, H., 2010. An information-theoretic framework for visualization. IEEE Trans. Vis. Comput. Graphics 16, 1206–1215.

Chi, E.H., 2000. A taxonomy of visualization techniques using the data state reference model. In: InfoVis'00, pp. 69–75.

Cover, T.M., Thomas, J.A., 2006. Elements of Information Theory. John Wiley & Sons.

Dinkla, K., Westenberg, M.A., van Wijk, J.J., 2012. Compressed adjacency matrices: Untangling gene regulatory networks. IEEE Trans. Vis. Comput. Graphics 18, 2457–2466.

Dork, M., Riche, N.H., Ramos, G., Dumais, S., 2012. Pivotpaths: Strolling through faceted information spaces. IEEE Trans. Vis. Comput. Graphics 18, 2709–2718.

Dunne, C., Riche, N.H., Lee, B., Metoyer, R., Robertson, G., 2012. Graphtrail: analyzing large multivariate, heterogeneous networks while supporting exploration history. In: CHI'12, pp. 1663–1672.

Dunne, C., Shneiderman, B., 2013. Motif simplification: Improving network visualization readability with fan and parallel glyphs. In: CHI'13, pp. 3247–3256.

Eger, S., 2013. Restricted weighted integer compositions and extended binomial coefficients. J. Integer Seq. 16.

Elmqvist, N., Do, T.N., Goodell, H., Henry, N., Fekete, J., 2008. Zame: Interactive large-scale graph visualization. In: PacificVis'08, pp. 215–222.

Elmqvist, N., Fekete, J.D., 2010. Hierarchical aggregation for information visualization: Overview, techniques and design guidelines. IEEE Trans. Vis. Comput. Graphics 16, 439–454.

Gansner, E., Koren, Y., North, S., 2004. Topological fisheye views for visualizing large graphs. In: InfoVis'04.

Gray, W., Bogovic, J., Vogelstein, J., Landman, B., Prince, J., Vogelstein, R., 2012. Magnetic resonance connectome automated pipeline: An overview. IEEE Purse 3, 42–48.

Hadlak, S., Schumann, H., Schulz, H.J., 2015. A survey of multi-faceted graph visualization. In: EuroVis'15 State-of-the-Art Reports.

Heer, J., Perer, A., 2011. Orion: A system for modeling, transformation and visualization of multidimensional heterogeneous networks. In: VAST'11, pp. 51–60.

Herman, I., Melancon, G., Marshall, M.S., 2000. Graph visualization and navigation in information visualization: A survey. IEEE Trans. Vis. Comput. Graphics 6, 24–43.

Jusufi, I., Kerren, A., Zimmer, B., 2013. Multivariate network exploration with jauntynets. In: IV'13, pp. 19–27.

Kang, H., Plaisant, C., Lee, B., Bederson, B.B., 2007. Netlens: Iterative exploration of content-actor network data. Inf. Vis. 6, 18–31.

Kerren, A., Purchase, H., Ward, M.O., 2014. Multivariate Network Visualization (Proc. Dagstuhl Seminar 13201). Springer.

Ko, S., Afzal, S., Walton, S., Yang, Y., Chae, J., Malik, A., Jang, Y., Chen, M., Ebert, D., 2014. Analyzing high-dimensional multivariate network links with integrated anomaly detection, highlighting and exploration. In: VAST'14, pp. 83–92.

Liu, Z., Navathe, S.B., Stasko, J., 2011. Network-based visual analysis of tabular data. In: VAST'11, pp. 41–50.

Lorrain, F., White, H.C., 1971. Structural equivalence of individuals in social networks. J. Math. Sociol. 1, 49–80.

MacQueen, J.B., 1967. Some methods for classification and analysis of multivariate observations. In: Proceedings of 5th Berkeley Symposium on Mathematical Statistics and Probability, Vol. 1, pp. 281–297.

Pretorius, A.J., van Wijk, J.J., 2006. Visual analysis of multivariate state transition graphs. IEEE Trans. Vis. Comput. Graphics 12, 685–692.

Pretorius, A.J., van Wijk, J.J., 2008. Visual inspection of multivariate graphs. Comput. Graph. Forum 27, 967–974.

Quigley, A., Eades, P., 2000. FADE: Graph drawing, clustering and visual abstraction. In: GD'00, pp. 197–210.

Shen, Z., Ma, K.L., Eliassi-Rad, T., 2006. Visual analysis of large heterogeneous social networks by semantic and structural abstraction. IEEE Trans. Vis. Comput. Graphics 12, 1427–1439.

Shi, L., Liao, Q., Sun, X., Chen, Y., Lin, C., 2013. Scalable network traffic visualization using compressed graphs. In: IEEE BigData'13, pp. 606–612.

Shneiderman, B., 1996. The eyes have it: A task by data type taxonomy for information visualizations. In: Proceedings of the IEEE Symposium on Visual Languages, pp. 336–343.

Shneiderman, B., Aris, A., 2006. Network visualization by semantic substrates. IEEE Trans. Vis. Comput. Graphics 12, 733–740.

Sindre, G., Gulla, B., Jokstad, H.G., 1993. Onion graphs: aesthetics and layout. In: Proceedings of the IEEE Workshop on Visual Languages, pp. 287–291.

Stasko, J., Görg, C., Liu, Z., 2008. Jigsaw: supporting investigative analysis through interactive visualization. Information Visualization 7, 118–132.

Takahashi, S., Fujishiro, I., Takeshima, Y., Nishita, T., 2005. A feature-driven approach to locating optimal viewpoints for volume visualization. In: IEEE Visualization, pp. 495–502.

Tang, J., Zhang, J., Yao, L., Li, J., Zhang, L., Su, Z., 2008. Arnetminer: Extraction and mining of academic social networks. In: KDD'08, pp. 990–998.

Tominski, C., Abello, J., Schumann, H., 2009. Cgvłan interactive graph visualization system. Comput. Graph. 33, 660–678.

van den Elzen, S., van Wijk, J.J., 2014. Multivariate network exploration and presentation: From detail to overview via selections and aggregations. IEEE Trans. Vis. Comput. Graphics 18, 2709–2718.

Viola, I., Feixas, M., Sbert, M., Groller, M.E., 2006. Importance-driven focus of attention. IEEE Trans. Vis. Comput. Graphics 12, 933–940.

Wang, C., Shen, H.W., 2006. Lod map-a visual interface for navigating multiresolution volume visualization. IEEE Trans. Vis. Comput. Graphics 12, 1029–1036.

Wang, C., Shen, H.W., 2011. Information theory in scientific visualization. Entropy 13, 254–273.

Wang, C., Yu, H., Grout, R.W., Ma, K.L., Chen, J.H., 2011. Analyzing information transfer in time-varying multivariate data. In: PacificVis'11, pp. 99–106.

Wang, C., Yu, H., Ma, K.L., 2008. Importance-driven time-varying data visualization. IEEE Trans. Vis. Comput. Graphics 14, 1547–1554.

Wattenberg, M., 2006. Visual exploration of multivariate graphs. In: CHI'06.

Weaver, C., 2010. Multidimensional data dissection using attribute relationship graphs. In: VAST'10, pp. 75–82.

White, D.R., Reitz, K.P., 1983. Graph and semigroup homomorphisms on networks of relations. Social Networks 5, 193–234.

Wu, Y., Takatsuka, M., 2006. Visualizing multivariate network on the surface of a sphere. In: APVis'06, pp. 77–83.

Xu, K., Cunningham, A., Hong, S.H., Thomas, B.H., 2007. Graphscape: integrated multivariate network visualization., In: APVis'07, pp. 33–40.