EgoNetCloud: Event-based Egocentric Dynamic Network Visualization

Qingsong Liu[*] Yifan Hu[†] Lei Shi[*] Xinzhu Mu[‡] Yutao Zhang[§] Jie Tang[§]

ABSTRACT

Event-based egocentric dynamic networks are an important class of networks widely seen in many domains. In this paper, we present a visual analytics approach for these networks by combining data-driven network simplifications with a novel visualization design - EgoNetCloud. In particular, an integrated data processing pipeline is proposed to prune, compress and filter the networks into smaller but salient abstractions. To accommodate the simplified network into the visual design, we introduce a constrained graph layout algorithm on the dynamic network. Through a real-life case study as well as conversations with the domain expert, we demonstrate the effectiveness of the EgoNetCloud design and system in completing analysis tasks on event-based dynamic networks. The user study comparing EgoNetCloud with a working system on academic search confirms the effectiveness and convenience of our visual analytics based approach.

1 INTRODUCTION

The egocentric network, as part of a larger network, is composed of a focal node (the "ego") and a set of nodes having direct connection to the ego (the "alters"), and all the edges between the ego and its alters, as well as among all the alters. In many scenarios, such an egocentric network serves as the foundation to understand the role of the ego in the full-scale network. For example, in email networks, the egocentric connections can help to infer one's communication type and pattern in the social interaction [13]. In mobile networks, the egocentric text-message conversation of the subscribers can be used to detect potential mobile spammers and promote appropriate responses [30]. Effective egocentric network visualizations have been shown to be critical as both the visual evidence for the role classification [25] and the exploratory tool to uncover hidden network patterns [21].

On small networks (e.g., below 100 nodes), the node-link diagram and the force-directed layout algorithm perform reasonably well. However, many networks are alive, i.e., they change over time. Depicting the temporal dynamics of an egocentric network can introduce extra visual clutter because of the proliferation of network edges [8, 21, 25]. Moreover, in many event-based networks, a clique sub-graph is produced for each event that occurred in the network, creating dense graphs that are hard to lay out and comprehend due to the edge crossing and visual clutter. In this paper, we consider the problem of visualizing event-based egocentric dynamic networks. We shall discuss in more detail in Section 3 that most dynamic networks are event-based, differentiated only by the number of related nodes on an event. One example is the co-authorship network of a researcher during his career. Each of his publications can be seen as an event. On each event, there can be a number of related nodes, aka the co-authors on the same publication. By portraying a clear, interactive image of his egocentric dynamic network built on the publications, we can provide a visual summary of his evolving academic collaborations.

Visualizing event-based egocentric dynamic networks is a non-trivial task. First, the visual design should reveal both the egocentric network structure and the temporal dynamics of the ego/alter nodes. This rules out a naive use of the node-link network diagram. Second, such a network can have a much larger size and complexity than a static egocentric network due to the dynamic and event-based nature. Applying traditional graph layout algorithms on the entire network can not satisfy the new requirements on the performance, visual metaphor, and the layout constraint. Third, interactions should be re-designed pertaining to these networks to enable the fine-grained visual exploration and analysis on temporal, topological and contextual aspect of the egocentric network.

In this paper, we propose a visual analytics based solution by combining novel data processing algorithms with an effective visual metaphor design. We have built an integrated visualization system called *EgoNetCloud*. Our major contributions are summarized as:

- *Data-driven empirical algorithms*, to prune, compress and filter event-based egocentric dynamic networks into smaller but more informative abstractions (Section 3);
- *EgoNetCloud visual metaphor and interactions*, to display and explore both the egocentric network structure and their temporal dynamics (Section 4.1, 4.2);
- *Fast and constrained layout computation*, to fulfill the requirement of the new visual metaphor and maintain fine readability (Section 4.3);
- *Comprehensive evaluations*, to demonstrate the effectiveness of the EgoNetCloud design through a user study comparing our system with a production system, and a real-world case study. (Section 5).

2 RELATED WORK

2.1 Network simplification

Increasing computational scalability and reducing visual complexity are among the key considerations in visualizing dynamic networks. One way to lower the visual complexity is through network simplifications in which the central idea is to prune edges or filter nodes. An extreme simplification that still keeps the graph connected can be obtained by a minimum spanning tree algorithm. The path-oriented simplification [27] removes edges that do not affect the quality of best paths between any pair of nodes. There are numerous other measures for edge importance. Girvan's edge betweenness [17] by the number of paths that run along the edge, and Birnbaum's component importance [7], defined as the probability that the edge is critical to maintain a connected graph.

As the size of network increases, plotting every node creates visual clutters, which hinders network understanding and analysis tasks. Apostolico [3] introduces a compression scheme based on the topological structure of the Web Graph that combines efficient storage with fast retrieval for the information on a node. The SEG [24] method visually condenses large network traffic graphs without sacrificing the connectivity information. Due to the intrinsic structures in these networks, SEG was shown to condenses some

[*]SKLCS, Institute of Software, Chinese Academy of Sciences, e-mail:{liuqs,shil}@ios.ac.cn. Lei Shi is the corresponding author.

[†]Yahoo Labs, New York, e-mail:yifanhu@yahoo.com.

[‡]Academy of Art and Design, Tsinghua University, e-mail:mxz12@mails.tsinghua.edu.cn.

[§]Department of Computer Science and Technology, Tsinghua University, e-mail:stack@live.cn, jery.tang@gmail.com.

graphs by more than 20 times while preserving the critical connectivity information. In this paper, we adopt the SEG method to compress the nodes in egocentric networks. All the aforementioned compression algorithms are for unweighted graphs. Toivonen [28] proposed methods for the compression of weighted graphs. While the compression is often based on the network connectivity, it can also be done by combining nodes with the same attribute [9]

2.2 Dynamic Network Visualization

While static graph visualizations are often divided into node-link and matrix representations, we identify the representation of time as the major distinguishing feature for dynamic graph visualizations. Dynamic graph visualization focuses on the challenge of the visual and computational complexity introduced by the extra time dimension. Many different visualization techniques have been introduced for dynamic graph structures which can be classified hierarchically [5]. As Beck discussed in [4], the time dimension can be mapped in an animation to a simulated time (time-to-time mapping), which creates an intuitive dynamic graph visualization. Ghani [16] found that node speed and target separation are prominent visual metrics for user perception of the animation. An alternative is to use a space dimension of the generated visualization to represent a timeline (time-to-space mapping). By showing the complete sequence of graphs in a static image [29], or as a diary [12], it promises to provide a better overview of the time and the changed among timesteps. Federico [11] proposed a visual analytic approach that supports a multi-faceted analysis of dynamically changing networks.

Traditionally, dynamic network visualization was studied especially on specific types of graphs such as trees [20] and directed acyclic graphs [22], drawn as a 2D node-link diagram. A matrix representation with time-line was proposed for visualization of genealogy data [6]. Visualization for egocentric network has so far motivated a variety of ideas. Farrugia et al. [10] proposed a tree-ring layout in which the time was encoded into multiple concentric circles from the ego node. The alters were replicated at each active time slot and placed equidistantly on the ring. In contrast, Shi et al. [25] proposed a 1.5D dynamic network visualization (1.5D), based on the egocentric data reduction of the dynamic network. It introduces a temporal glyph to represent the trend of the ego node. The time information of each of these edges is encoded by the location of the edge's endpoint on the trend glyph. Compared to the 1.5D approach, our *EgoNetCloud* introduces simplification methods to deal with large-scale networks, and adopts an improved stress majorization approach which optimizes the topological and temporal context with reference to the ego node.

2.3 Graph Layout Algorithm

Graph visualization seeks to find a visual representation of a graph that captures structures and reveals anomalies in the graph. Node-link diagrams and adjacency matrices [5] are two basic ways of representing graphs, with the former being more widely used. The graph layout problem is to find the most appropriate node positions for a node-link representation of a graph. In doing so, often a certain objective cost is optimized [19]. Two basic layout algorithms are force-directed placement and stress optimization [14, 15]. A robust way to solve the stress model is stress majorization [15]. For our application, we need to place nodes close to their ideal positions in time and space, while maintaining readability and reflecting the topology of the graph. We propose a technique which modify the stress majorization process to achieve these goals.

3 Event-based Egocentric Dynamic Network

3.1 Definition

Consider the *dynamic network* represented by a time-varying graph $I = (V, E)$ spanning a time period $[0, \Gamma)$. The graph consists of a node (vertex) set V and an edge (link) set E. Each edge $e \in E$ (node $v \in V$) is associated with a time set $T(e)$ ($T(v)$), which contains multiple discrete time points or continuous time periods defining the activation time of the edge (node). An example is given in the middle column of Figure 1. It is assumed that the underlying graph I is simple, i.e. there is no loop edges or multiple edges between two different nodes. Both directed and undirected graphs are allowed, but for simplicity, we assume that I is undirected. We use the notation $u \leftrightarrow v$ to denote that nodes u and v are connected by an undirected edge.

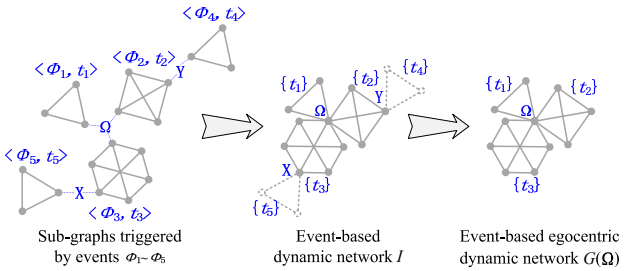


Figure 1: Construct an event-based egocentric dynamic network from a set of events and their associated edges.

The *egocentric dynamic network* $G(\Omega) = (V(\Omega), E(\Omega))$ with regard to the ego node Ω is essentially an induced subgraph of the full-scale dynamic graph I, as shown in the right column of Figure 1. Its node set includes the ego node Ω, and all nodes that Ω is connected with, $V(\Omega) = \{\Omega\} \cup \{v | v \leftrightarrow \Omega\}$.

The graph $G(\Omega)$ is said to be *event-based* if any discrete time point (or continuous time period) $t \in T(e)$ of the edge $e \in E(\Omega)$ is associated with an activation event ϕ, denoted by $e \sim< \phi, t >$. For example, a co-authorship of two researchers at time t is exactly triggered by one of their joint papers published at t. Conversely, the event-based egocentric dynamic network $G(\Omega)$ can be uniquely constructed from the event set Φ. Each event $\phi \in \Phi$ will add an activation of the edge e at time t if $e \sim< \phi, t >$, as illustrated in the left column of Figure 1. In a co-authorship egocentric network, each paper publication will introduce a clique subgraph among all authors of the paper, activated at the publication time.

In fact, every dynamic network can be seen as event-based if each activation of a particular edge is considered a single event. For example, establishing a friendship tie in online social networks corresponds to a "connect" event. Sending a mobile short message belongs to a "contact" event.

3.2 Analysis Framework

The main goal of EgoNetCloud is to reveal both the structure and temporal dynamics of event-based egocentric dynamic networks. This calls for a series of processing steps starting with the full-scale network normally in a large size. In extracting the egocentric dynamic network, we split the ego node into multiple sub-nodes. Each of the sub-node represents a discrete time point or continuous time period. As such, the static egocentric network can be decomposed into the dynamic network according to each edge's timestamp. However, the resulting egocentric dynamic network $G(\Omega)$ may still contain a large number of nodes and edges. Displaying all nodes and edges can result in significant visual clutters, while judiciously removing some nodes and edges, or at least summarizing them, often give a more succinct view of the network without significant loss of information. With the above principle in mind, our network processing pipeline can be summarized as below:

- *Edge Pruning*: to remove low-weight edges to simplify the egocentric dynamic network. For example, in academic collaboration networks, the complete graph of each collaboration could be converted into a backbone spanning connected subgraph:

one that retains important collaboration relationship, while contains much less number of edges (Section 3.3).

- *Node Compression*: to group nodes with the same connection pattern together. This can greatly reduces the number of nodes, especially effective after edge pruning is applied. (Section 3.4).
- *Graph Filtering*: to reduce nodes and their related edges by certain rule-based policy, so as to reserve key roles and connections in the visualization (Section 3.5).

The resulting system takes a JSON formatted file with "node" and "link" object arrays. A node object contains the attributes of each node (e.g., its id and label); a link object contains the attributes on link (e.g., a time stamp), as well as the ids of the end nodes.

3.3 Edge Pruning

In a large complex network, including all the edges in the network visualization often severely reduces the clarity of display and conceals alters' role. Pruning unessential edges thus helps to highlight key nodes and edges in the network.

In academic networks, a complete co-authorship graph, which is traditionally used to indicate the paper collaboration, has a lot of redundancy. For example, consider one paper with three authors in this order: A, B and C. Suppose A has collaborated many times with B but none with C. Being a student of B, C can also published a lot with B. In this case, the edge between A and C is less important and can be pruned to display only substantial collaborations. To quantify this idea, we compute edge weights on each event-based clique graph, using information from the full-scale network, then pruning edges in the egocentric network with the weight smaller than a threshold.

3.3.1 Computing Edge Weights

Take the academic network case as an example, we consider papers as documents and authors as words, and compute author-author similarity in a way analogous to computing the word-word similarity in text analysis. Specifically, we form the sparse matrix M of dimension $m \times n$, where m is the number of papers, and n the number of authors. The (i, j)-th entry of the matrix, m_{ij}, is non-zero if author j is an author of paper i.

For each paper (a row of matrix M) with, say, n authors, we can either make each nonzero row entry as $\frac{1}{n}$, or use a credit allocation algorithm to decide the weight for each author. As an example, suppose we have 3 papers involving 5 authors represented by the following matrix

$$M = \begin{pmatrix} 1/3 & 1/3 & 0 & 1/3 & 0 \\ 1/3 & 1/3 & 0 & 0 & 1/3 \\ 1/4 & 1/4 & 1/4 & 0 & 1/4 \end{pmatrix}$$

This matrix says that papers 1 and 2 have 3 co-authors each, paper 3 has 4 coauthors. For example, paper 5 is written by author 2 and 3. We calculate the edge weight between two authors j and l as cosine similarity of j-th and l-th columns of the matrix M.

Recency based scaling: so far we assumed that each paper contributes to the similarity equally if the number of authors are the same. Considering that a recent collaboration is more important that a collaboration many year ago, we scale the contribution from each paper by the inverse of its age, thus contribution of paper k to the similarity of authors i and j is scaled by $\gamma_k = \frac{1}{\text{ageOf}(k)+s}$, where s is a parameter. A joint paper s year old is 1/2 as important as a recent paper ($\gamma_k = 1/(s+s)$ vs $1/s$).

Scaling based on author ordering: up to now we have been calculating author similarity using purely historical information. However in a paper co-authored by A, B and C, just because B and C have stronger a historical collaboration record than A and B does not mean that in the current paper, the edge between B and C is more important than that between A and B. To look at this in another way, if A is the leading author, and all pairs of authors have the same historical tie, then a reasonable null hypothesis is that the links between A and B, and A and C, should be stronger than the link between B and C, because A is leading the collaboration. Therefore for a paper k involving authors i and j, the similarity between authors i and j due to paper k is scaled by $\sigma_{kij} = 1/r(k,i) + 1/r(k,j)$, where $r(k,i)$ and $r(k,j)$ are list orders of author i and j. So overall, the similarity along an edge e between authors i and j is

$$w(e) = \frac{\sum_{k=1,\ldots,m} \gamma_k^2 \sigma_{kij} m_{ki} m_{kj}}{\left(\sum_{i=1,\ldots,m} \sigma_{kij} (\gamma_k m_{ki})^2\right)^{1/2} \left(\sum_{j=1,\ldots,m} \sigma_{kij} (\gamma_k m_{kj})^2\right)^{1/2}}$$

Note that this is a weighted cosine similarity, with weights $\gamma_k \sigma_{kij}^{1/2}$.

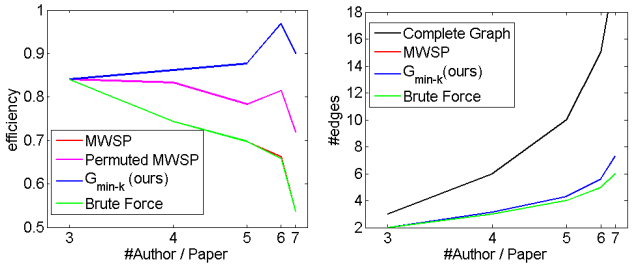
So far we assumed that authors are ordered by their contributions. But in some academic fields authors are always listed in alphabetical order. In such a case we use a credit allocation algorithm [23] to capture the coauthors contribution to a publication as perceived by the scientific community, and leave out σ_{kij} in $w(e)$.

3.3.2 Pruning the Edges

After computing the weights of each edge in a complete paper collaboration graph, we propose an algorithm to prune the edges. The algorithm has three objectives: (1) prune as many edges as possible; (2) retain important edges; (3) preserve the connectivity of the graph. The simple method such as keeping the top k heaviest weighted edges may not work, because of the possibility to disconnect the resulting subgraph.

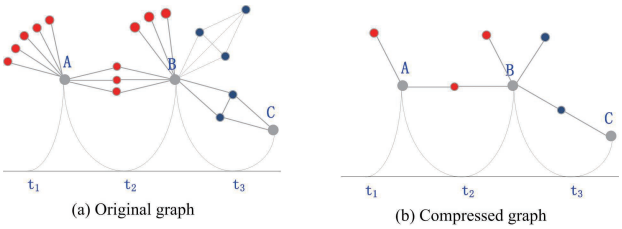
To measure how well important edges are kept after applying a particular pruning algorithm, we propose the follow metric. Give a weighted graph $G = \{V, E\}$ with $w(e)$ the edge weight, we define the efficiency of a sub-graph $G_s = \{V_s, E_s\}$ as $\text{efficiency}(G_s) = \frac{\sum_{e \in E_s} w(e)}{\text{sum of top } |E_s| \text{ edge weights in } E}$.

Initially we attempted to use the maximum weighted spanning tree (MWSP) algorithm to generate a tree, and pruned all non-tree edges. This does satisfy considerations 1) and 3), however our experiment results below show that it does poorly in retaining the important edges (consideration 2)). In our experiment we took 1791 TVCG papers from year 1995 to year 2013 with no more than 7 co-authors (we exclude papers with more than 7 co-authors because the sample size for such papers is too small for reliable statistical averaging). We then apply pruning algorithms to these author networks and plot the efficiency as a function of the number of co-authors in Figure 2(a). For comparison with MWSP, we took the same weighted complete graph, and permute the edge weights, then generate the maximum weighted spanning tree. In the figure, the red line represents the efficiency of MWSP, while the magenta line the



(a) (b)

Figure 2: The performance of four edge pruning algorithms on the co-authorship network of 1791 TVCG papers, 1995∼2013: (a) algorithm efficiency; (b) number of edges after pruning. The X axis represents the bins by the number of co-authors per paper. The interval between two bins is proportional to the number of papers in the starting bin.



(a) Original graph (b) Compressed graph

Figure 3: An example for the node compression: (a) Several red and blue nodes have the same neighborhood connectivity. They can be compressed to reduce the graph size; (b) The final compressed graph with grouped nodes.

efficiency of the permuted MWSP. If the backbone of author networks have a natural tendency to be tree like, we would expect that MWSP/permuted MWSP should capture a larger share of the edge weights (have high efficiency) on a real author network than on a random network. However Figure 2(a) shows that the opposite is true. In other word, the backbone of a real author network does not have a propensity to be a tree. This is understandable, because in a paper co-authored by, say, 6 people, there could well be subgraphs of authors who form tight cliques, due to previous collaborations.

Therefore instead of the spanning tree, we propose to use the smallest connected maximum weighted spanning subgraph. This graph, denoted as G_{min-k}, is defined as the smallest possible connected subgraph consists of the top weighted edges. In other word, no edge in the complement of this subgraph has weigh greater than any of the edges in G_{min-k}, furthermore, taking away the edge(s) with the lowest weight in G_{min-k} would make it disconnected.

As we can see from the blue line in Figure 2(a), the average efficiency of G_{min-k} is much higher than the both the maximum spanning tree, and the maximum spanning tree with randomly permuted weights. It captures most of the important edges. At the same time, as Figure 2(b) demonstrated, the number of edges in G_{min-k} isn't much more than the maximum weighted spanning tree. This means that G_{min-k} prunes a large amount of unimportant edges, while maintains the backbone of the author network, and preserves the connectivity of the graph. To verify the effectiveness of G_{min-k}, we compare it to the "brute force approach" (BF) edge pruning algorithms from [31], which iteratively picks the edge whose removal best keeps the connectivity. As Figure 2(a) show, the results of BF are almost the same to the maximum spanning tree.

3.4 Node Compression

Edge pruning simplifies the graph by reducing the number of edges, while node compression simplifies by reducing the number of nodes. The main idea of the node compression method is to merge nodes with the same or similar connectivity. Nodes with the same connectivity are known as "structurally equivalent", and can be identified in linear time [24] and merged. After node compression, the ultimate graph consists of two kinds of nodes: single nodes remaining from the original graph (solid blue disk in the Figure 4) and the meta-node grouped from multiple sub-nodes in the original graph (disk with blue outer ring in the Figure 4).

We describe the node compression algorithm below. Given an egocentric dynamic network, let W be the graph adjacency matrix. We devise a two step approach: in the first step, the diagonal elements of W is set to 0, allowing the grouping of nodes with exactly the same connectivity, such as the red nodes in Figure 3, to be merged. In the second step, the diagonals are set to 1 and all the original nodes that have the same connectivity and are also linked to each others are grouped, as the blue nodes in Figure 3. This basic compression is lossless. In addition, we employ a fuzzy compression that not only compresses nodes with the same connectivity, but also nodes with highly similar connectivity. When combined with the edge pruning process discussed earlier, the edge pruning not only reduces the edge number, but also results in better node compression. Taking Jie Tang's academic collaboration egocentric dynamic network as an example, the raw graph has 235 nodes, with the edge pruning step, it can be compressed into 156 nodes, while without edge pruning it can be only compressed into 198 nodes.

3.5 Graph Filtering

After the edge pruning and node compression, the egocentric dynamic network may still have excessive number of nodes and edges which create cluttered visualizations. To resolve this problem, we provide two classes of node filters on alter nodes. The first is based on the connectivity of alters with the ego in the egocentric dynamic network, and the second based on the active time of alters.

Using graph connectivity, we calculate the importance of a node using the PageRank algorithm on the egocentric network. In addition, we provide filtering according to the degree of the nodes, as well as based on time period. For application on the author collaboration network, we also provide node filtering based on the total number of citations, both in the overall network, and in the egocentric dynamic network.

4 EgoNetCloud

4.1 Visualization Design

On the temporal and structural analysis of egocentric dynamic networks, the single-view static visualizations [18, 25] have been shown to be more effective than using the network animation [29] and the small multiples [10]. In a 1.5D design [25], a trend graph is introduced to display the temporal dynamics of the egocentric network, while the node-link graph is preserved to represent the network connections. This design, however, has drawbacks when applied to the event-centric scenario. First, the egocentric network in our case can grow to a much larger size and complexity due to the event definition and the clique subgraphs. The existence of the standalone trend graph reduces the available space for the network layout and leads to more edge crossings. Second, the temporal and categorical dynamics on the event can not be displayed in the original design. In this paper, we introduce EgoNetCLoud, a new visualization for event-centric egocentric dynamic networks. The main innovation lies in: (1) a hybrid design that places the egocentric network inside the trend graph, to save the layout space and reduce edge crossings; (2) a double-sided cloud metaphor to replace the trend graph to display the temporal dynamics of both the egocentric network and its central event in multiple types.

We describe the visual design of EgoNetCloud through the deployment on the ArnetMiner (AMiner) data set [1], a popular academic search and social networking website. We focus on the egocentric network of Prof. Jie Tang (the inventor of AMiner), where his paper publications are considered to be the central events (aka the ego nodes). Till April 2013, Prof. Tang's egocentric network has expanded to 235 co-authors and 3709 relationships triggered by 160 papers. After the processings in Section 3, this network is simplified to 158 nodes and 395 edges.

Figure 4 illustrates the EgoNetCloud system interface on Prof. Tang's egocentric network. The main EgoNetCloud visualization is given in Figure 4(b). The height of the trend graph above the horizontal line represents the dynamics on the number of co-authors collaborating with Prof. Tang over time. Inside this cloud, the simplified version of his egocentric network is displayed. Each node in the network (aka the alter node) is drawn in a filled blue disk, indicating one single or group of his co-authors. For the group nodes, a hollow circle is drawn outside the disk to indicate the group nature. The size of the disk shows the importance of the co-author (group) in the egocentric network. A few importance measures can be applied by setting in the control panel (Figure 4(a)). Between the nodes, network edges are drawn to represent the co-authorship relationship among the alters. Three display modes of the network edge

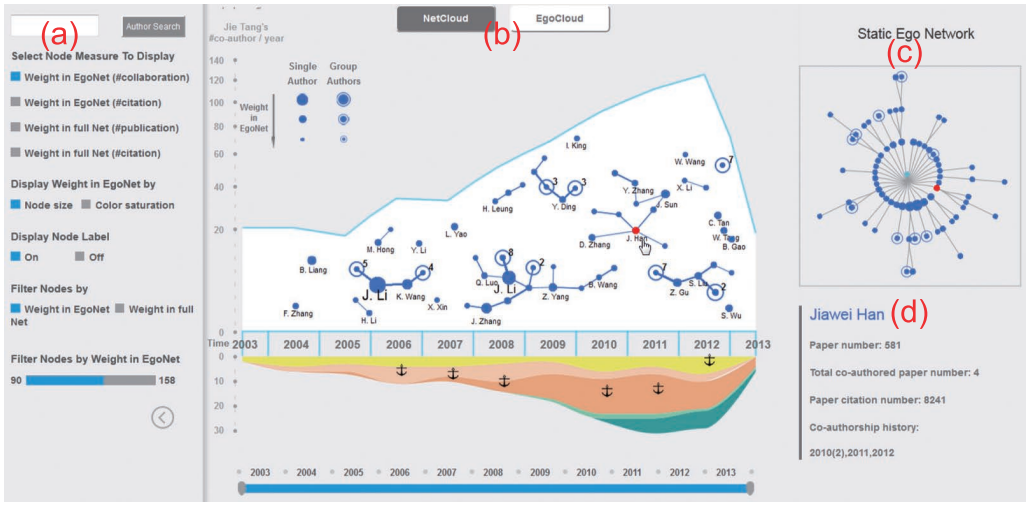
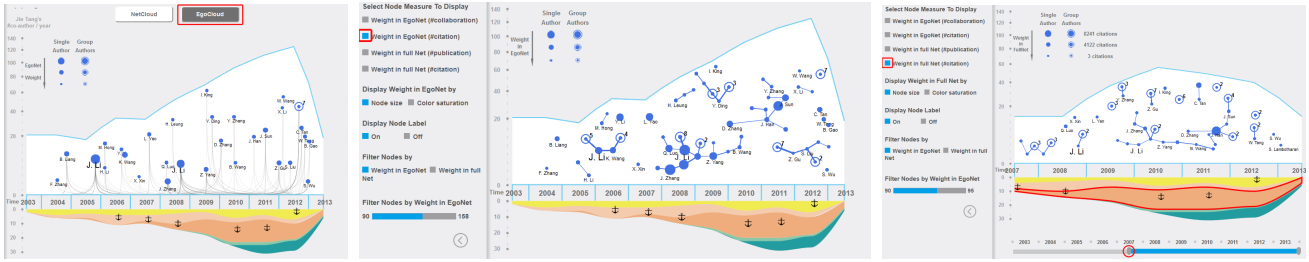


Figure 4: The EgoNetCloud system interface of Prof. Tang's egocentric network from AMiner: (a) the control panel in the Left; (b) the main EgoNetCloud visualization in the center; (c) the static egocentric network view in the top-right; (d) the detail panel in the bottom-right. The NetCloud mode is selected in this figure, so that only the connections among alter nodes are drawn.



(a) (b) (c)

Figure 5: EgoNetCloud interactions: (a) EgoCloud mode, only the connections between Prof. Tang and his co-authors are drawn; (b) display the number of joint citations with Prof. Tang by the node size of the alter; (c) drill down to the network on data mining from 2007 to 2013.

can be configured: the NetCloud mode (Figure 4(b)) that only displays the edges between the alter nodes, in straight lines; the EgoCloud mode (Figure 5(a)) which only displays the edges between the alter nodes and the ego node, in curved and bundled lines; and the full mode when all edges are displayed.

Below the horizontal line of Figure 4(b), the stacked trend graphs indicate the number of papers published by Prof. Tang over time (i.e., the events), which is split by the topic of the paper (e.g., data mining, semantic web), and filled in different colors. Inside these trend graphs, several anchor icons are placed in certain positions to represent the high-citation papers on each topic. In Figure 4(c), a static view aggregating the egocentric network over time is displayed in a radial layout, which works as an overview to the egocentric network, where users select a group of interesting co-authors to check their dynamics in the main EgoNetCloud visualization.

4.2 Interaction

After an interview with the management team of AMiner, we summarize four interactive analysis tasks they expect from any advanced visualizations. *T1.* Find out one's key collaborators. *T2.* Find out one's core teams. *T3.* Find out one's representative papers. *T4.* Find out one's major research topics and how they evolve over time. To fulfill these tasks, we have designed rich interactions on EgoNetCloud. They are described in a typical user trail in analyzing Prof. Tang's egocentric network.

The user starts from the overview in Figure 4. To identify Prof. Tang's key collaborators (*T1*), he can switch to the EgoCloud mode in Figure 5(a) by clicking the toggle buttons on top of the visualization. Both the node size and the node degree in this graph indicates the number of collaborations with Prof. Tang. The top two researchers are found to be two "J. Li" in separate locations. By hovering the nodes, their full names are discovered as "Juan-zi Li" and "Juanzi Li". In fact, they are the same person, Prof. Juanzi Li in the same lab with Prof. Tang, who removed the short dash from her publication name around the year of 2008. Back to Figure 4(b), the user can locate 8∼10 nontrivial subgraphs, which indicate the teams collaborating with Prof. Tang. To find out which teams are core to Prof. Tang's performance (*T2*), the user specifies in the control panel (Figure 4(a)) to map the total number of citations an author (alter) received together with Prof. Tang (ego) to the node size. This is shown in Figure 5(b), where three core teams in large node sizes are discovered. Two have Prof. Li in the center, another has Prof. Sun as the key people. These findings were confirmed to be true by checking with Prof. Tang in person.

On *T3*, the high-citation papers are directly shown as the anchor icons inside the event trends below the horizontal line in Figure 4(b). To get more details, the user can hover one icon to highlight the team working on the corresponding paper. To study Prof Tang's research topics (*T4*), the user can examine the stacked event trends below the horizontal line in Figure 4(b). Two topics seem to dominate Prof. Tang's research, the semantic web (pink) before 2007 and data mining (orange) after 2007. The user can drill down to one topic by clicking on the corresponding event trend. He can also filter the data by time with the range selector at the bottom of the main EgoNetCloud visualization. In Figure 5(c), only the authors collaborating with Prof. Tang in the data-mining topic from 2007 to 2013 are displayed. The node size is switched to represent the number of citations each author received in the entire career. The name of Prof. Han showed up as the key senior people in the data mining community that ever collaborated with Prof. Tang.

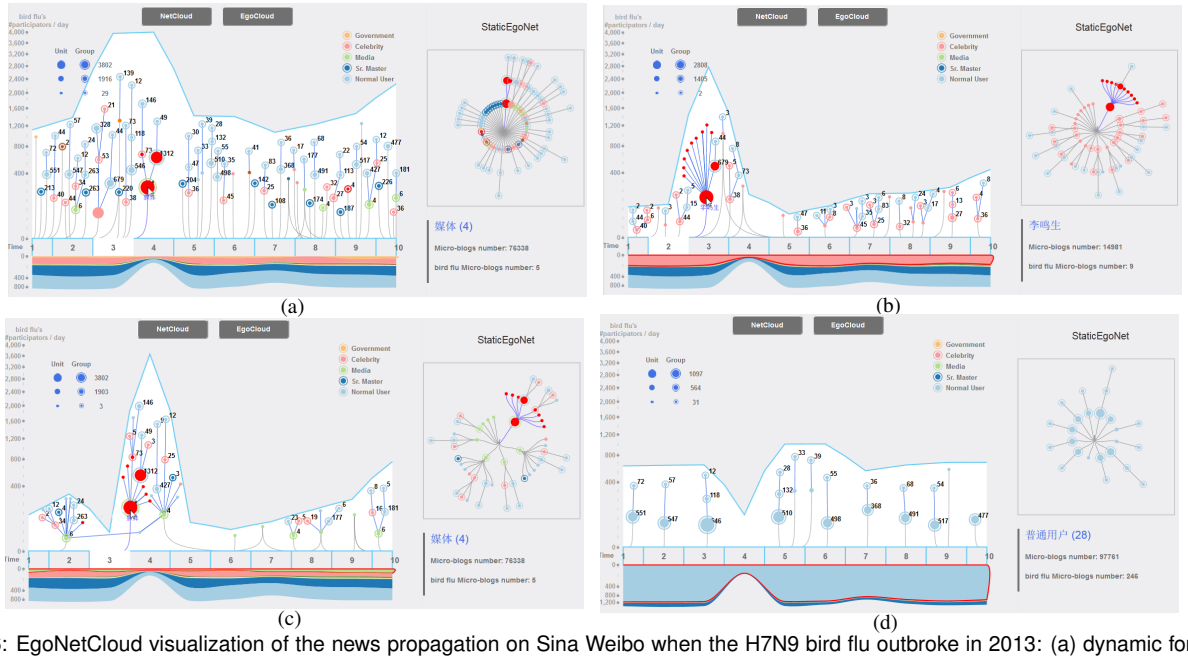


Figure 6: EgoNetCloud visualization of the news propagation on Sina Weibo when the H7N9 bird flu outbroke in 2013: (a) dynamic forwarding tree raised by all types of users; (b) the forwarding pattern initiated by the celebrity; (c) the forwarding pattern raised by the media in the follow-up; (d) the activities of ordinary users. The nodes are colored in green for media, pink for celebrity, blue/light-blue for the advanced/ordinary user.

4.3 Layout Algorithm

The position of the alter nodes in the 2-dimensional space is of special significance to our design. Given the egocentric dynamic network, the final layout should satisfy our design rationale. Take the AMiner scenario as an example, the co-authors that collaborated frequently should be placed close to each other, and horizontally, each node should be close to the average time point of its collaboration events with the ego. Additionally, the nodes should be well separated to avoid visual clutter.

We first set the initial position for all the nodes, $\overline{X} = \{\overline{X}_1, \ldots, \overline{X}_n\}$, according to the alter's interaction time and frequency with the ego. However, for an ego node having multiple interactions over a short time, even with our filtering process, there may still be a high concentration of nodes in certain part of the cloud space. To make full use of the cloud space, we divide the cloud into several parts according to time periods, relocating nodes according to each part's size. Then, mapping the nodes in the canvas into the cloud space according to the ratio of the cloud boundary height and the canvas height.

After setting the initial positions, we compute the final layout by taking into account the ideal locations of the nodes, as well as ensuring that the final layout fully reflect the connectivity of the graph. We can not simply apply the standard force-directed layout algorithm [14] or the stress majorization algorithm [15] to layout the graph, because neither takes into account position constraints. Therefore we propose a constrained stress majorization approach. The original stress model attempts to achieve graph-theoretic target distances, by minimizing the stress energy $\sum_{i<j} w_{ij}(\|X_i - X_j\| - d_{ij})^2$. Here, the distance d_{ij} is the graph-theoretical distance between node i and node j. The weights w_{ij} equals d_{ij}^{-2}. While the model does not take into account the desired location of the nodes, we actually want to derive a layout that constrains the nodes to be near their ideal position, and at the same time reflects the connectivity of the graph. We minimize the stress function $\alpha \sum_{i<j} w_{ij}(\|X_i - X_j\| - d_{ij})^2 + (1-\alpha)\sum_{i\le n}\|X_i - \overline{X_i}\|^2$

We expand this function to $\alpha \sum_{i<j} w_{ij}\|X_i - X_j\|^2 - 2\alpha\sum_{i<j}\delta_{ij}\|X_i - X_j\| + (1-\alpha)\sum_{i\le n} X_i^2 - 2(1-\alpha)\sum_{i\le n} X_i\overline{X_i}$, where $\delta = w_{ij}d_{ij}$ Since $\sum_{i<j}\delta_{ij}\|X_i - X_j\|$ can be bounded by the Cauchy-Schwartz inequality and the rest are quadratic, so a lower bound for the stress function is defined as $\alpha Tr(X^T L^w X) - 2\alpha Tr(X^T L^Z Z) + (1-\alpha)Tr(X^T I(X - 2\overline{X}))$. The minimum of this quadratic function is achieved when

$$(\alpha L^w + (1-\alpha)I)X = \alpha L^Z Z + (1-\alpha)\overline{X} \quad (1)$$

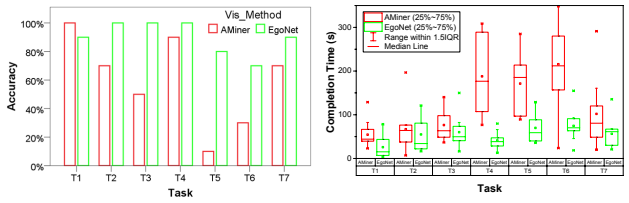
Following the stress majorization process [15], we start with $Z = \overline{X}$, and solve (1) for X. We then replace Z by the solution X. We repeat this process until convergence.

5 Evaluation

5.1 Case Study for Online Social Networks

The EgoNetCloud system has been demonstrated to work well on academic collaboration networks (Section 4.1), here we showcase its usage on general-purpose social networks. We consider Sina Weibo [2], the largest Chinese microblogging website, known as the Twitter in China and providing exactly the same service. In Mar. 2013, a new subtype of the bird flu, code name H7N9, was detected on human in mainland China, which eventually caused more than a hundred confirmed human cases and 21 deaths. This news was quickly distributed on the social media and created great panics in the general public. We study the propagation of the bird flu news on Sina Weibo using EgoNetCloud, the event-based network visualization system. Note that unlike the co-authorship network, the forwarding graph of each microblog forms a tree structure. There is no need for the edge pruning in our analysis framework.

We collected all the messages on Sina Weibo matching the bird flu related keywords or having the relevant topic tag, within one month period from the time of the outbreak. In total, there are 30,000 messages and 20,000 users, in both the original and the forwarded microblogs. We parsed the sending time of each message and their forwarding tree for visualization. The initial EgoNetCloud view in the first ten days is shown in Figure 6(a), where the egocentric event is defined as all the original messages on this topic. The trend line in the top illustrates the number of users forwarding bird flu related messages. In the bottom, the multiple trend lines depict the number of original messages published by each type of users. A clear pattern of the event rise and fall is observed: on the



(a) Accuracy (b) Completion Time

Figure 7: The user performance of EgoNetCloud v.s. ArnetMiner.

third and fourth day, the event reaches the first peak by the number of participants. The scale of the event is greatly amplified by the message forwarding (retweets) in that the number of original messages shown as the height of the bottom trends is much smaller than the number of forwarded messages, shown as the height of the top trend. There is an exception on the fourth day when few original microblog is published. After manually checking the raw data, we found that most original messages on that day have been deleted by the site administrator.

In Figure 6, we display the dynamic message forwarding trees by compressing users according to their publication date and the account type on the Sina Weibo. Note that Sina Weibo has applied a certification mechanism to classify the real-life identity of registered users as their account types. Besides the ordinary user without a certification, the account type includes: the celebrity, the mass media, the enterprise and the advanced user having paid to the website. Each node in the EgoNetCloud stands for one group of users, with the node size representing the number of their original+forwarded messages on the bird flu topic. The user/account type is shown by the node color (green for media, pink for celebrities, blue for advanced users and light-blue for ordinary users). Two nodes are connected if the messages from the user group of the first node are dominantly forwarded by users in the second node. The second node is placed at the top of the first node in the vertical direction, so that the forwarding tree pattern is revealed. In the bottom trends of EgoNetCloud, the original messages are split into multiple colored stacks according to the author's account type.

An overview picture is displayed in Figure 6(a) to analyze the overall trend of the outbreak, including the related users in all account types, their messages and forwarding relationships. However, this picture does not given a clear indication on which group of users are the most influential in the discussion. We then apply the topic-based interaction in EgoNetCloud to only display the forwarding network initiated by celebrities, as shown in Figure 6(b). In the same vertical scale with Figure 6(a), it can be found that the initial rise of the event on the third day is mostly contributed by the celebrities. On the detailed network inside the cloud metaphor, the celebrity's messages are frequently forwarded by the ordinary and advanced users on either the original or celebrity-forwarding messages (pink nodes are connected to the blue nodes above them). In another view, by clicking the topic trend for the media in Figure 6(c), it is shown that the peak on the fourth day is mainly contributed by the forwarding of the media's messages, while ordinary and advanced users are the main driving force in intensifying the discussion. Lastly, we look at the activity initialized by the ordinary user, as shown in Figure 6(d). It is interesting to notice that though ordinary users account for the largest number of users and messages (the light blue trend in the bottom), their messages are not forwarded as much as the celebrities and the media. Instead, their trend is relatively smooth and does not exhibit any abrupt peak.

In summary, by analyzing the event-based egocentric network on Sina Weibo, we can find that the discussions there on the Mar. 2013 bird flu event is driven by different types of users in a three-stage manner: at first, the celebrities initiated the discussions, then the media relayed on the story, and finally, the ordinary users continued to talk about the topic even after the peak.

5.2 User Study on EgoNetCloud and AMiner

We conducted a controlled user study to compare the performance of the EgoNetCloud system with the AMiner [26]. We selected AMiner as the baseline system because most other relevant websites (e.g., DBLP, Microsoft Academic Search) implement a similar design in displaying the publication profile of a researcher. AMiner enjoys an advantage in our scenario by integrating an egocentric network visualization, see Prof. Tang's home page on AMiner as an example (http://cs.aminer.org/person/jie-tang-1458619.html).

Experiment design. We applied a between-subject design in this study to eliminate the learning effect. Twenty subjects were recruited, who are graduated students majoring in computer science or visualization. Ten subjects started from Figure 4, the default view for Prof. Tang by EgoNetCloud. The other ten subjects started from Prof. Tang's home page in the AMiner website. Note that all the subjects know little about Prof. Tang's research work, so their performance largely depend on the experiment system in use.

Each subject was asked to complete seven tasks expanded from Section 4.2 and answer the following questions in a best-effort manner. A six-minute deadline is set for each task. *T1*: Which year (2003∼2013) did Prof. Tang publish the most papers? (name one) *T2*: Who has collaborated the most times with Prof. Tang in terms of the number of papers? (name one) *T3*: In Prof. Tang's egocentric network, who has collaborated the most times with Prof. Juanzi Li? (name one) *T4*: In which years did Prof. Tang collaborate with Prof. Jiawei Han? (name three) *T5*: Which researchers have the most overall citations in Prof. Tang's egocentric network? (name two) *T6*: Which researchers have the most citations together with Prof. Tang? (name two) *T7*: Who has collaborated the most times with Prof. Tang in the data mining field (name one)? These tasks are designed according to an interview with the management team of AMiner, and revised after pilot studies with another group of subjects. The tasks can be categorized into three classes: (1) the temporal information related (T1); (2) the egocentric network related (T2, T3, T5, T6); (3) a combination of the two (T4, T7).

Before the test session, we introduced a training session to make sure that each subject understood all tasks well and got familiar with the EgoNetCloud/AMiner system interfaces. The training session included three sample tasks from each task type with another researcher as the ego node. The organizer checked the answer of each training task and explained any ambiguity on the task immediately. In the test session, we recorded the subject's answer and the completion time in each task. The task completion time was measured after the subject read the question, so that the reading skill variation was factored out.

Result. We collected 140 entries (20 subjects $\times$ 7 tasks) on both the user's answer and their completion time. These results were analyzed separately on each task, by comparing the group using EgoNetCloud with the group using AMiner. The significant level was set at 0.05 throughout the analysis.

Accuracy: The accuracy of each answer is calculated as the percentage of correctness, by comparing with the ground truth. Figure 7(a) illustrates the average accuracy on each task. It is shown that, except T1 (90% v.s. 100%), EgoNetCloud achieves higher task accuracies than AMiner on all the other tasks. We conducted binary logistic regressions to capture the boolean value of the task accuracy. It is shown that, the contribution of the choice of system to the task accuracy variation is statistically significant ($p < .001$), when all tasks are considered together. The EgoNetCloud system increases the likelihood (odds) of correctly answering each task to 6 times of that of AMiner (95% CI = [2.4, 15]). The goodness of fit of the logistic regression is 0.176 (Nagelkerke R Square).

Completion Time: The distribution of completion times on each task is summarized as grouped box plots in Figure 7(b). It can be found that on all tasks, the subjects answered questions faster with the EgoNetCloud system than with the AMiner system. Applying

the one-way ANOVA test, the difference is significant on T1 ($p = 0.04$), T4 ($p < 0.001$), T5 ($p < 0.001$) and T6 ($p = 0.001$).

Summary and Discussion. The user study results demonstrate that, on several important egocentric network analysis tasks, EgoNetCloud performs better than the baseline AMiner system. On the task related to the temporal information only (T1), our system is faster, but does not have an advantage in accuracy, nor a big disadvantage. On the egocentric network related tasks (T2, T3, T5, T6), EgoNetCloud achieves the largest performance gain in both efficiency and accuracy, especially on hard tasks (T5, T6). On tasks combining temporal and egocentric network information (T4, T7), EgoNetCloud improves over the baseline on the efficiency, though the task accuracy is not increased significantly.

Through the user and case studies, we find that the single-view static visualization design of EgoNetCloud enjoys several advantages in real-life analysis tasks. Most notably, the capability to answer advanced user questions by combining temporal, structural and contextual information together into the same egocentric network visualization. The rich interactions also help a lot in drilling down to details to answer specific questions. On the other hand, such a design does bring overheads in certain aspects and scenarios. First, compared with the list and simple visualization based system, the training time for users to work well with EgoNetCloud is much longer. Some users find the initial view to be too information-dense. They suggest to apply more aggressive network simplification settings. Second, EgoNetCloud is currently designed to visualize dynamic networks with all the data set pre-processed offline. No support is provided for the streaming dynamic network, which can be more important to study the real-time behavior through networks, such as the information diffusion on social media.

6 Conclusion

In this paper we presented EgoNetCloud, a visual design as well as an interactive data visualization system for displaying and exploring both the network structure and the temporal dynamics of event-based egocentric networks. We proposed and employed a number of algorithms to prune, compress and filter these networks to reduce visual clutters, and to reveal the salient part of the network. A novel constrained stress majorization algorithm computes a graph layout for the egocentric network within the cloud metaphor. The resulting network display can exhibit both the relationship between the ego and the alters in the time dimension, and the connection among alters, while maintaining a good graph readability. Case studies on the co-authorship network and the online social network show that EgoNetCloud is able to uncover network patterns and help to complete user tasks, in a way that is more effective than existing text-based discovery systems. A controlled user studies confirmed that users achieve better performance with EgoNetCloud than a baseline system on both the task efficiency and accuracy. We are now working towards making the EgoNetCloud system available, initially as a visual interface for online academic search engines.

Acknowledgements

This work is supported by China 973 project 2014CB340301, NSFC projects (No. 61379088, 61222212), National Social Science Foundation of China (No. 13&ZD190). We thank Dr. Linjia Xu from college of humanities social sciences, UCAS, for providing the microblog data set.

References

[1] ArnetMiner dataset. http://arnetminer.org/billboard/citation.

[2] Sina Weibo. http://weibo.com.

[3] A. Apostolico and G. Drovandi. Graph compression by BFS. *Algorithms*, 2(3):1031–1044, 2009.

[4] F. Beck, M. Burch, and S. Diehl. Towards an aesthetic dimensions framework for dynamic graph visualisations. In *IV'09*, pages 592–597, 2009.

[5] F. Beck, M. Burch, S. Diehl, and D. Weiskopf. The state of the art in visualizing dynamic graphs. In *EuroVis 14–State of The Art Reports*, pages 83–103, 2014.

[6] A. Bezerianos, P. Dragicevic, J. Fekete, J. Bae, and B. Watson. GeneaQuilts: A system for exploring large genealogies. *IEEE Trans. Vis. Comput. Graph.*, 16(6):1073–1081, 2010.

[7] Z. W. Birnbaum. On the importance of different components in a multicomponent system. Technical report, DTIC Document, 1968.

[8] M. Burch, C. Vehlow, F. Beck, S. Diehl, and D. Weiskopf. Parallel edge splatting for scalable dynamic graph visualization. *IEEE Trans. Vis. Comput. Graph.*, 17(12):2344–2353, 2011.

[9] C. Dunne and B. Shneiderman. Motif simplification: improving network visualization readability with fan, connector, and clique glyphs. In *CHI'13*, pages 3247–3256, 2013.

[10] M. Farrugia, N. Hurley, and A. Quigley. Exploring temporal ego networks using small multiples and tree-ring layouts. In *ACHI'11*, pages 79–88, 2011.

[11] P. Federico, W. Aigner, S. Miksch, F. Windhager, and L. Zenk. A visual analytics approach to dynamic social networks. In *i-KNOW'11*, pages 1–8, 2011.

[12] J.-D. Fekete. GraphDiaries: Animated transitions and temporal navigation for dynamic networks. *IEEE Trans. Vis. Comput. Graph.*, 20(5):740–754, 2013.

[13] D. Fisher. Using egocentric networks to understand communication. *IEEE Internet Computing*, 9(5):20–28, 2005.

[14] T. M. Fruchterman and E. M. Reingold. Graph drawing by force-directed placement. *Software: Practice and experience*, 21(11):1129–1164, 1991.

[15] E. R. Gansner, Y. Koren, and S. North. Graph drawing by stress majorization. In *GD'04*, pages 239–250, 2004.

[16] S. Ghani, N. Elmqvist, and J. S. Yi. Perception of animated node-link diagrams for dynamic graphs. In *Computer Graphics Forum*, volume 31, pages 1205–1214, 2012.

[17] M. Girvan and M. E. Newman. Community structure in social and biological networks. *PNAS*, 99(12):7821–7826, 2002.

[18] S. Hadlak, H.-J. Schulz, and H. Schumann. In situ exploration of large dynamic networks. *IEEE Trans. Vis. Comput. Graph.*, 17(12):2334–2343, 2011.

[19] Y. Hu. Algorithms for visualization large networks. In *Computational Scientific Computing*, pages 525–549. CRC Press, 2012.

[20] S. Moen. Drawing dynamic trees. *IEEE Software*, 7(4):21–28, 1990.

[21] C. W. Muelder, T. Crnovrsanin, A. Sallaberry, and K.-L. Ma. Egocentric storylines for visual analysis of large dynamic graphs. In *IEEE BigData*, pages 56–62, 2013.

[22] S. C. North. Incremental layout in dynadag. In *GD'95*, pages 409–418, 1995.

[23] H.-W. Shen and A.-L. Barabási. Collective credit allocation in science. *PNAS*, 111(34):12325–12330, 2014.

[24] L. Shi, Q. Liao, X. Sun, Y. Chen, and C. Lin. Scalable network traffic visualization using compressed graphs. In *IEEE BigData*, pages 606–612, 2013.

[25] L. Shi, C. Wang, Z. Wen, H. Qu, C. Lin, and Q. Liao. 1.5 d egocentric dynamic network visualization. *IEEE Trans. Vis. Comput. Graph.*, 21(5):624–637, 2015.

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

[27] H. Toivonen, S. Mahler, and F. Zhou. A framework for path-oriented network simplification. In *IDA'10*, pages 220–231. 2010.

[28] H. Toivonen, F. Zhou, A. Hartikainen, and A. Hinkka. Compression of weighted graphs. In *KDD'11*, pages 965–973.

[29] B. Tversky, J. B. Morrison, and M. Betrancourt. Animation: can it facilitate? *International journal of human-computer studies*, 57(4):247–262, 2002.

[30] C. Wang, Y. Zhang, X. Chen, Z. Liu, L. Shi, G. Chen, F. Qiu, C. Ying, and W. Lu. A behavior-based SMS antispam system. *IBM Journal of Research and Development*, 54(6):1–16, 2010.

[31] F. Zhou, S. Mahler, and H. Toivonen. Simplification of networks by edge pruning. In *Bisociative Knowledge Discovery*, volume 7250 of *LNCS*, pages 179–198. 2012.