egoComp: A Node-link Based Technique for Visual Comparison of Ego-network

Dongyu Liu, Fangzhou Guo, Bowen Deng, Yingcai Wu, and Huamin Qu

Fig. 1. The figure shows the egoComp system: (A) provides a list for selecting different network datasets to analyze. (B) displays node id of two selected egos. (C) offers three buttons for users. When different buttons are clicked, corresponding layout algorithm for right side ego-network will be implemented. (D) shows the graph overview, which is drawn by Huyifan algorithm[9]. (E) presents the view for ego-network comparison.

Abstract—Analysis of ego-networks is a critical research problem when analyzing social networks, as an ego-network represents the social circle a person actually contact with. One of the core tasks in ego-network analysis is visual comparison, which includes edge comparison and node comparison. Although various works have been done to support comparing normal networks and ego-networks, intuitive node comparison of two ego-networks is still challenging. In this paper, we propose egoComp, an intuitive and expressive visualization technique, to analyze the node difference between two ego-networks. To preserve the latent structure of ego-network and lay emphasis on intuitiveness, our design is node-link based (radial tree layout) and uses side-by-side method to make comparison. We are the first to apply storyflow-like links into node-link graph to reveal the relation of two ego network in individual node detail. Also, three different layout algorithms, including origin, greedy and assignment algorithm, are presented for users so that they can decide which one to use according to their demands. In addition, limitations are discussed thoroughly and corresponding potential solutions are also given. Finally we list some significant and promising future work.

Index Terms—ego-network, node comparison, node-link, side by side, visual analytics

1 INTRODUCTION

An ego-network is a kind of special social networks. Unlike common social networks, they have a particular central node designated as ego. Other nodes in the network are direct or indirect neighbours of the ego. An ego-network shows the relations among the ego and its neighborhood. Important insights can be obtained by comparing ego-networks of different entities when exploring and analyzing social networks. In a social network, the closeness of two people can be represented by the similarity of their ego-networks. For example, in an academic collaboration network, different collaboration mode can be revealed by comparing two authors’ ego-networks.

The difference between two ego-networks has two aspects, which are node difference and connectivity difference. Node difference reveals the difference of social circle, for example, people who have different occupation, age, or gender have different friends. Connectivity difference shows the different patterns of entity relations, for example, an intermediary’s alters may not have links as they possibly don’t know each other, while a professor’s alters have many links as they are in the same social circle and know each other.

We present a technique called egoComp which focuses on the node comparison of two ego-networks, which solves following questions:

Q1: Which nodes are both friend nodes in the two ego-networks?
Q2: Which nodes have the same level in the two ego-networks?
Q3: Which nodes have the different level in the two ego-networks?

Three major ways of visual comparison are concluded in [5], including juxtaposition, superposition, and explicit encodings. Combining different visualization forms, such as node-link diagram and adjacency matrix, with different comparison techniques, researchers have developed various graph comparison systems. Although these systems also support comparing ego-networks by treating the network as normal networks, the layout and comparison method don’t fit the ego-network data because ego-network latently contains a tree structure. Thus, we developed an intuitive and interactive ego-network comparison system. We firstly applied juxtaposition (etc. side-by-side) of two transformed node-link diagram, of which each represents a ego-network. To decrease the user’s memory cost, storyflow-like [12] links are used to link the nodes that appear in both ego-networks. The layout of nodes are calculated by an optimization algorithm to avoid the crossing of the links.

The major contributions of this work are:

- An ego-network comparing technique, which is based on side-by-side node-link diagrams and visual links. To make the visualization intuitive, storyflow-like links are applied to show the similarity of two ego-networks.
- Two layout optimization algorithms, a greedy algorithm and an assignment algorithm. We evaluated and compared the complexities of the two algorithms.

The paper is structured as follows: Section 2 presents related work. Section 3 lays out details of our approach. Section 4 discusses the performance and limitations of our approach. Conclusion and future work are presented in section 5.

2 RELATED WORKS

Ego-network is a critical research topic in social network analysis. [7, 16, 20] use node-link diagram to visualize the ego-networks. Depending on the data and the tasks, the layout of node-link diagram can be modified, such as in [20], a time line is added to visualize the dynamic networks, and in [16], nodes with specific distance to the ego are arranged on the circle with corresponding radius. In our work, we use the similar layout as in [16] but we apply a special layout algorithm to arrange the nodes on semicircles with least link crossings.

Bremer et al. [3] concludes that comparing is one of abstract visualization tasks. Visual comparison techniques are classified into three categories: juxtaposition, superposition, and explicit representation of the relationships [5]. Juxtaposition designs present objects side-by-side and consume analysts’ memory to make connections between objects. Superposition designs overlay objects on top of others. Explicit encodings calculate the relationships between objects and visualize these relationships.

Some systems support single type of comparison techniques, such as juxtaposition of node link diagrams [15, 16], superposition of node link diagrams [2], and superposition of adjacency matrix [1]. Others support multiple types of techniques, such as [4, 6]. Merging networks together requires layout adjustment and more encodings to support visual comparison, which makes visualization less intuitive. VisLink [4] links the same objects together in two different visualizations. However, the links are not well arranged and it causes severe clutter. Besides, the layout of the compared visualization are fixed. Our work inherits the idea of VisLink but avoids the clutter problem by using Storyflow-like links in our work and optimizing the layout of the ego-network. Edge bundling algorithms are used to solve the clutter problem caused by the links in [8, 22], but the number of edges is vanished after bundling.

Evaluating the similarity of network data is another way to compare two networks. By MDS or other projection algorithms, a network can be represented by a dot in scatter plot [13] and the distance between two dots represents the similarity of two networks. Similarity-based comparison method describes difference macroscopically buts it lacks details of compared objects. Thus it is less intuitive than our technique.

Ogawa and Ma [18, 21] present Storyline to visualize the evolution of software. This technique is efficient to show the change of relationships of objects and is implemented to analyze different datasets, such as storyflow [12]. We combine the storyflow with node-link diagram to show the similarity of two ego networks.

3 EGO COMPARISON

3.1 System Overview

Two datasets at different level of scalability are used in our system. One is Les Miserables [10] with 77 nodes and 254 edges, describing the coappearance network of characters in novel Les Miserables. The other one is NetScience [17] with 1589 nodes and 2742 edges, illustrating the coauthorship network of scientists working on network theory and experiment. As shown in Fig. 1, our system interface mainly consists of two views, the overview and the comparison view. After users select a dataset (Fig. 1-A), graph will be drawn below the overview (Fig. 1-D), by using Huyifan Layout Algorithm [9], to leave users a comprehensive understanding of this selected dataset. Besides, by using mouse wheel, users are able to zoom-in and zoom-out the view to further explore the graph in different level of detail. Nodes in the overview can be chosen as ego by mouse clicking. When two egos are selected, users can click the buttons shown in Fig. 1-C to choose the comparison layout algorithms and then see the result in comparison view.

3.2 Visual Design

This subsection will define some important terms in this paper and introduce the visual encoding in comparison view.

![Fig. 2. The ids of left ego and right ego are respectively 9912 and 9881. The layout algorithm is arranging alters on both LEN and REN by their BFS order, and corresponds to the "origin" button in Fig. 1-C. (A) The left side ego-network, is abbreviated as LEN. (B) Ego. (C) Hop1 Alter. (D) Hop2 Alter. (E) Link. (F, G) Alter Angle, from 0 to π. (H) Session. LEN & REN. The ego-network on the left side is abbreviated as LEN, and the one on the right side is abbreviated as REN. Ego. Users can pick two different nodes in the overview as the central nodes of ego-network in comparison view, the first selected one for LEN, the second one for REN. Central nodes in ego-network are designated as ego. The left ego and right ego are respectively coloured with red and purple to visually distinguish them. Hop1 alter. The ego’s directly connected friends are called hop1 alter, and they are assigned uniformly on inner-semicoloncircle. Hop2 alter. The hop1 alters’ directly connected friends, are called hop2 alter, and they are assigned uniformly on outer-semicoloncire. In this paper, we focus on analyzing ego’s hop1 and hop2 alters. Node color. LEN’s hop1 alters, which are also REN’s hop1 or hop2 alters(common friends), are encoded with color orange. And LEN’s hop2 alters, which is common friends of REN, are encoded with color blue. Node size. The size of nodes stands for the strength of relation with ego. The larger the size is, the stronger the relation between this node and ego is.](image-url)
Link. The link between LEN’s alter and REN’s alter illustrates they are the same node in the overview.

Alter angle. The included angle between perpendicular line and the line trough ego and this alter, where the value of angle is from 0 to π (from top to bottom).

Session. The 6 rectangles in the middle area are called sessions. Similar to the storyflow[12], links will flow into specific sessions and flow out according to their attribute. For the left top session, links starting from alters whose angle less than 1/3π will flow into it. For the right top session, links ending in alters whose angle less than 1/3π will flow through it. The middle and bottom four session are similar to top two.

3.3 Layout Algorithm

It’s obvious that the alter positions in REN will greatly influence the link crossing numbers. To reduce the number of crossing, we need to use algorithm to position the alters. We assume the layout of LEN is given by simply using Breadth First Search, (i.e., BFS). It means that the node visited earlier by BFS visiting order, will be put in the position with smaller angle (from top to bottom). What we need to compute is the layout of REN’s alters. There are three layout algorithms for REN’s alters, where the default one is exactly the same as the one we use in LEN layout. The other two layout algorithms are dedicated to reduce the sum of alter angle difference (Eq. 1), because lower sum of alter angle difference means decreasing of the number of crossing of links. Following I will describe these two algorithms.

\[ \sum_{i=1}^{n} (\alpha_i - \beta_i) \]  

(1)

Where \( n \) is the number of common alters, \( \alpha_i \) means \( i \)th common alter’s angle in LEN, and \( \beta_i \) means \( i \)th common alter’s angle in REN.

3.3.1 Greedy Algorithm

We propose n very intuitive greedy algorithm to reduce the sum of alter angle difference. It’s obvious that the alter layout in REN’s hop1 and hop2 can be dealt with separately. The core idea is that, for each common alter, whose position in LEN is given, we greedily pick the appropriate position on REN such that the alter angle difference is minimal. The time complexity is \( \Omega(n^2) \). The specific process is shown in Algorithm 1.

Algorithm 1 Greedy Algorithm

1: function GREEDY\_LAYOUT(hopLevel)
2:   commonAlters ← CommonAltersinputhopLevelofREN
3:   positionSet ← AllAltersinputHopLevelofREN
4:   sortByAngle(commonAlters)
5:   for alter ∈ commonAlters do
6:     selectedPosition ← findMinimal(alter, positionSet)
7:     layout[alter] ← selectedPosition
8:     positionSet ← positionSet − alter
9:   end for
10: return layout
11: end function

3.3.2 Assignment Problem

Actually, we can convert this problem into assignment problem[14], more specifically, to find a minimum weight perfect matching in a weighted bipartite graph. Since the alter layout in REN’s hop1 and hop2 can be computed separately, we will take how to figure out the layout of hop2 alters for an example. For the sake that uncommon alters will not impact the link crossing, we only consider the common alters. Suppose there are \( n \) common alters in hop2 of REN, and \( p \) hop2 alters of REN in total, meaning that there are \( p \) places we can put the alters in. Therefore, each common alter can be assigned in one of the \( p \) positions, and each position only accepts one alter. Since our purpose is to minimize the sum of alter angle difference, we are able to conduct this problem into assignment problem, a fundamental combinatorial optimization problem, which can be solved by Kuhn-Munkres algorithm[11] in \( O(n^3) \) and get the optimal result.

4 DISCUSSION

We suppose that the smaller the sum of alter angle difference is, the better visual effect it will achieve. As shown in Table 1, we select some pairs of alters as sample to test the performance of algorithm. We may easily see that in most cases, the result is improved hugely when the layout algorithm is applied. Although greedy algorithm will sometimes get local optimal results, even worse than the original layout (i.e., BFS) in few cases, it still performs as well as Kuhn-Munkres algorithm in most time. Fig. 3 provides a specific example to illustrate the difference, whose original layout is shown in Fig. 2. In fact, it should be noticed that the low sum of alter angle difference doesn’t represent the small number of crossing of links. Therefore, it is appropriate to enable users to choose layout algorithms by themselves according to their own flavors.

4.1 Performance

When the number of common alters of two ego-networks increases to hundreds, the number of links will exceed hundreds, and therefore
lead to a serious problem of visual clutter. To alleviate the visual clutter caused by links, we are considering to use sankey diagram[19] to bundle the lines through a same session into a entire flow. As to the nodes, techniques like heat-map or node collapse is a potential solution. Furthermore, more interactions can be added to highlight the individual node-link-node path according to users’ demands. We will leave it as our future work.

EgoComp is designed for comparing two ego-networks for the use of juxtaposition, so one of the limitations is to compare more than two ego-networks at the same time. Actually, when making comparison among multiple ego-networks, we don’t have to show so many details as comparing between only two ego-networks. Therefore, it’s possible to come up with a new design without such many details, in order to compare among multiple ego-networks.

Although greedy algorithm and Kuhn - Munkres algorithm can decrease the number of crossing of links effectively, they are inevitably to harm the original structure of ego-network, and make the alter layout itself less meaningful than the original layout(i.e., BFS). To balance the layout’s meaningfulness and cleanliness, our method is to provide different choices of layout algorithms for users to let them freely decide by themselves in line with their own demand.

Although links occupy lots of visual space, they are less informative and only illustrate which alter in LEN connect to which alter. In addition, the purpose of using sessions in the middle area is to visually classify alters by their angle attribute and make the view visually clean, however, angle attribute is not original network data’s own property. As to this, we don’t make full advantages of these visual space. To have the session convey more significant information included in network data, we are considering to judge which sessions the links should flow through according to link type(i.e., from hop1 alter to hop1 alter, from hop1 alter to hop2 alter, from hop2 alter to hop1 alter, from hop2 alter to hop2 alter). If doing this so, we should redesign the layout algorithms. This is left as our future work too.

5 Conclusion

In this paper, we mainly proposed egoComp, a visualization technique based on node-link diagram and juxtaposition, which aims to facilitate comparing between two ego-networks. EgoComp allows users to quickly choose two nodes in a graph as egos, and visualize the ego-networks expanded by these two egos by using radial layout with hop1 alters in inner-semicircle and hop2 alters in outer-semicircle. The storyline-like[12] links are applied to illustrate the alters’ position change in left ego-network and right ego-network. To reduce the sum of alter angle difference(Eq. 1) and make the view visually clean and symmetric, two optimization algorithms(i.e. greedy algorithm and Kuhn - Munkres algorithm[11]) are proposed. By doing some experiments, we find that the performance of greedy algorithm approaches Kuhn - Munkres algorithm in most time. However, for the sake of destructive effect of original ego-network structure, sometimes users are more willing to choose the original layout(i.e., BFS), especially when the number of common alters is very small.

Apart from the future work mentioned in section 4.2, there are still some other useful and significant work remains to be done in the future: Firstly, to facilitate users to choose the right pair of nodes as egos to compare, a new feature of guiding them to select nodes in overview is required. For example, we can provide visual clues for users to tell them what kind of roles are these nodes in overview playing, like the node with largest number of hop1 alters. Or for users who have selected one ego and want to choose another one with most similar ego-network to the selected one, we can also visually tell them by some assistant visual design. Additionally, in our current design, nodes are distributed uniformly along 1D arc with specific radius, which may lead to the loss of information about inner clusters existing in hop1 or hop2 itself. If we position nodes in 2D annulus area rather than 1D arc, it’s promising for us to find inner clusters in hop1 and hop2.

References