Improving the Readability of Clustered Social Networks using Node Duplication

Nathalie Henry, Anastasia Bezerianos, Jean-Daniel Fekete

Improving the Readability of Clustered Social Networks using Node Duplication

Nathalie Henry, Anastasia Bezerianos, and Jean-Daniel Fekete

Abstract—Exploring communities is an important task in social network analysis. Such communities are currently identified using clustering methods to group actors. This approach often leads to actors belonging to one and only one cluster, whereas in real life a person can belong to several communities. As a solution we propose duplicating actors in social networks and discuss potential impact of such a move. Several visual duplication designs are discussed and a controlled experiment comparing network visualization with and without duplication is performed, using 6 tasks that are important for graph readability and visual interpretation of social networks. We show that in our experiment, duplications significantly improve community-related tasks but sometimes interfere with other graph readability tasks. Finally, we propose a set of guidelines for deciding when to duplicate actors and choosing candidates for duplication, and alternative ways to render them in social network representations.

Index Terms—Clustering, Graph Visualization, Node Duplications, Social Networks.

1 Introduction

Social networks analysis is becoming increasingly popular with online communities such as FaceBook, MySpace or Flickr, where users log in, exchange messages or pictures, and generally interact with friends or collaborators. Many sociologists perform statistical analyses on these networks, attempting to fit their data to existing models according to *a priori* hypotheses. However, visual analysis becomes more and more popular as it offers the opportunity to view the raw data in an exploratory manner, without bias or *a priori* formed analysis questions, and opens new perspectives for previously analyzed datasets.

The most popular visual representation of social networks is node-link diagrams where persons (actors) are represented as nodes and their relationships as links. An important task when analyzing these networks is examining communities — i.e. groups of actors tightly connected to each other — to understand their relationship and roles. For example, communities clearly emerge when visualizing the coauthorship network of a conference, where nodes are researchers and links represent articles co-signed by other researchers. Different patterns are visible, such as student-advisor relationships or laboratory collaboration relationships. Studying these communities and patterns helps us understand for example who are the most connected/prolific authors and the boundary of their communities. When such communities are identified (for example by clustering algorithms), they can be grouped visually and structurally in a *clustered graph* (Figure 1a). This community representation suffers from two problems:

Clustering ambiguity: When an actor is connected to two or more communities: 1) most clustering algorithms place the actor in one of the communities (Figure 2 left,center); 2) others place shared actors between their communities, solving the unique assignment problem but increasing link crossings when several nodes belong to several communities; and 3) in few clustering algorithms, communities that share actors are visualized as overlapping clusters, increasing the visual complexity of the graph by introducing node overlap and link crossings due to the tight space packing. Visualizing the overlapping nodes is difficult or even impossible when the number of intersections increases.

- *Nathalie Henry is with INRIA-LRI and Univ. of Sydney, E-mail: nathalie.henry@lri.fr.*
- *Anastasia Bezerianos is with NICTA, E-mail: a.bezerianos@nicta.com.au.*
- *Jean-Daniel Fekete is with INRIA, E-mail: jean-daniel.fekete@inria.fr.*

Manuscript received 31 March 2008; accepted 1 August 2008; posted online 19 October 2008; mailed on 13 October 2008.
For information on obtaining reprints of this article, please send e-mail to: tvcg@computer.org.

Readability: When two communities share many connections, their links intersect and cross several nodes, hindering the identification of the particular actors connected. We qualify layouts with numerous overlapping nodes and edge crossings as having a high *visual complexity*.

Recently, a hybrid representation called NodeTrix [17] improved graph readability by representing clusters as visual adjacency matrices (Figure 1b). As actors are placed linearly on the four sides of the matrix, node overlapping is suppressed and intra- and inter-community edge crossing is reduced. However, when several actors connect two communities, the visualization still suffers from edge-crossings, reducing its readability. Moreover, deciding where to place an actor shared between communities remains a challenge.

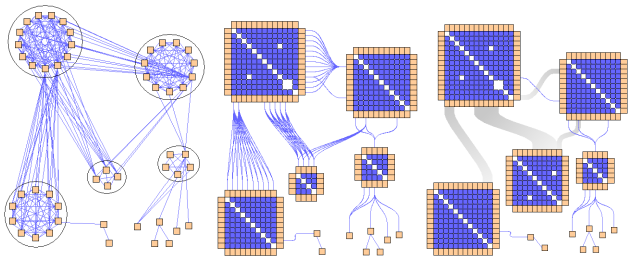
To improve readability and solve ambiguous clustering, we propose using actor duplications: introducing new nodes to represent aliases of actors. We implement this idea in NodeTrix. Figure 1 shows a NodeTrix representation before (b) and after (c) duplication. Actor duplication provides flexibility by allowing clustering of shared actors in multiple communities (Figure 2) and may reduce the networks' visual complexity by suppressing links. Nevertheless, it alters the network structure and may affect the user's perception during analysis. This article discusses ways of representing duplications and, through a controlled experiment, examines how duplicating actors affects social networks analysis.

2 Related work

Social networks describe persons or organizations (actors) and their relationships. Examples include genealogical trees, disease transmission and communication networks. In this paper, we concentrate on visual analysis techniques to analyze social networks (detailed information on statistical and structural methods can be found in [31]).

Exploring social networks involves tasks at several levels [27]. Low-level readability tasks include: finding if two actors are connected, how many actors lie between them, or if they are connected to the same set of actors. High-level tasks [31] include: *identifying communities* (C), i.e. cohesive groups of actors that are strongly connected to each other; *identifying central actors* (CA), i.e. actors either linked to many other actors, or bridging communities together; *analyzing roles and positions*, i.e. interpreting groups of actors (positions) and connection patterns (roles).

Higher level tasks are performed through completion of low-level tasks. For example, if linked actors have a large number of common connections that are themselves interconnected, they most likely form a community. Thus, social network analysis is closely related to network visualization [18] and graph drawing [8], and is affected by graph readability issues. However, the decomposition of high-level tasks into



(a) Node-Link (b) NodeTrix (c) Duplications

Fig. 1: Clustered graph representation of the same portion of a co-authorship network. Orange marks represent authors, blue co-authorship and grey duplication links. (a) Clustered node-link diagram with communities circled. (b) NodeTrix representation with communities being adjacency matrices (authors are placed in rows and columns; blue squares indicate co-authorship). Missing intra-community links appear. (c) NodeTrix where actors shared among communities are duplicated in each. Missing inter-community links appear.

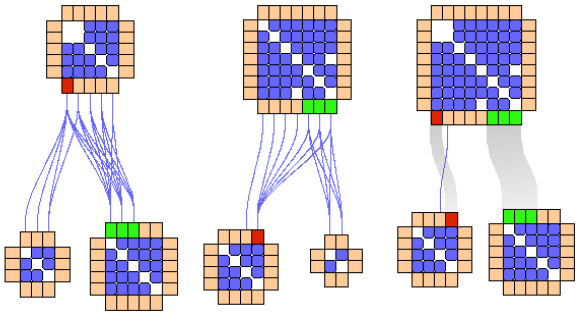


Fig. 2: Ambiguous Clustering: actors shared among communities in red and green are ambiguously placed in only one community (left, center), or using duplication in all their communities (right).

low-level tasks is very dependent on the actual visualization system: its visual representations, layout algorithms and provided interactions. Ideally, a good visualization system should allow most important tasks to be performed quickly and effortlessly through interaction and visual scanning, to help users gain insights on the social network they study.

A large number of systems exists for visually analyzing and representing social networks, the vast majority of which are based on node-link diagrams: 54 out of 55 according to INSNA [1]whereas visualcomplexity [2]lists more than 50 systems only using node-link diagrams. The readability of such diagrams has been studied by several authors: McGrath et al. [24] investigated the role of spatial arrangement of nodes in a node-link diagram and showed that node positions strongly affect the identification of central actors and communities. Other factors such as *node overlapping* or *edge crossing* affect the readability of node-link diagrams. Purchase [28] and Ware et al. [30] showed that edge bends, crossings number and angles affect the completion of connectivity tasks such as finding the shortest path.

Recent systems adopted other strategies to overcome node-link diagram readability problems: NetLens [21] and PaperLens [22] abandoned node-link diagrams for simple interactive visualizations, like bars or tables. Albeit effective in performing visual queries on network attributes, they do not adequately reveal topological properties required to perform high-level tasks such as finding communities. Others merge or combine multiple network representations. TreePlus [23] uses a tree layout centered on a node to visualize parts of a network without node overlapping or edge crossing. While effective for connectivity tasks (find paths), performing high-level tasks such as finding communities or central actors is difficult. MatrixExplorer [15] combines node-link diagrams and visual adjacency matrices to improve readability, but its authors underline the potential cognitive cost of switching back and forth between the two representations.

2.1 Clustered graph representation

Clustered graph representations reduce the visual complexity of graphs by grouping nodes, and are well suited to social networks as they highlight important structures, like communities and central actors linking them. A large body of methods [20] exists to automatically identify communities in social networks (computing clusterings). In many cases, algorithms give different clusterings and human interpretation is needed to find a consensus. Social analysts often need to adjust the level of clustering or edit the groups a posteriori. Eades et al. proposed several solutions to draw clustered graphs [11], eventually showing several levels of clustering [10] at the same time. Recently Ask-GraphView [2] used aggregated graph representation to navigate within very large networks. These representations are particularly effective for small-world networks [32], a common category of social networks that are globally sparse but locally dense, i.e. communities are linked by a few connections and actors.

However, these representations suffer from lack of readability when clusters are connected with many edges. As nodes are grouped close together, it is hard to avoid edge-crossing or to ensure crossings with clear angles. Holten proposed a method to bundle edges together [19], improving global readability for tasks like determining the two most connected clusters, but losing detailed connectivity information, such as seeing if two nodes belonging to two different clusters are connected. An alternative approach aggregates the links between clusters [3], only displaying a single link when any node in one cluster is connected to any node in another. Visual complexity is thus reduced, but detailed connectivity is no longer visible.

Recently, Henry et al. [17] proposed merging node-link and matrix representations with NodeTrix. NodeTrix represents clusters as visual adjacency matrices: each node is placed as a column and row in a matrix and links between nodes are marked in the matrix. Note for example in Figure 1b, how the diagonal is "empty" of links, as nodes are not linked with themselves. Since nodes and their links are placed in a matrix, node-overlapping is suppressed and readability is improved inside the cluster. Outside, incoming edge crossing is minimized, as there are four incoming alternatives for each node (2 per column and row). However, *ambiguous clustering* remains unsolved. Auber et al. [3] envision a clustering in which some central actors are extracted in the center of the graph. However, this is based on human interpretation and cannot be done automatically. Although it only partially addresses *ambiguous clustering*, if many actors that are member of several communities are extracted, the number of edges dramatically increases (extracted actors are connected to almost every actor in all communities), *degrading the representation readability*.

2.2 Information visualization and duplications

To improve readability and cluster ambiguity, we propose the *duplication of actors*. In the literature the concept of "duplication" is not clearly defined. Several terms exist: duplicates, clones, mirrors or aliases. In graph drawing, duplication is named *vertex splitting*.

The presence of non-identified duplicates in a dataset is generally considered as noise. It might skew statistics and provide misleading information. However, when appropriately introduced into a network representation, duplications can improve its readability or ease its exploration. For example, Eades and Mendoca [9] show that duplicating nodes can reduce edge-crossing and reveal symmetries, two important aesthetic criteria in graph drawing. In information visualization, examples are scattered among the literature where duplication appears among features of a system with few details given, if any. We particularly noticed duplication use in tools visualizing graphs as trees such as OntoRama [12] or TreePlus [23]. Thanks to duplications, graphs can be presented as trees by suppressing cycles. In these examples, duplicated elements are displayed using a specific color and, in TreePlus the

[1]http://www.insna.org

[2]http://www.visualcomplexity.com

user can click on an element to see its aliases highlighted. Other examples include genealogical tree visualization systems such as GenoPro [1]. Here, duplications are used as a presentation or exploration tool: for example by duplicating married couples, members of each side of the family can analyze their own family sub-tree. In GenoPro, a dash line links the orginial node to its alias.

While duplications are particularly used in systems visualizing trees or graphs represented as trees, to our knowledge, they have never been used or tested in clustered graph representations.

3 Duplication variations and designs

As social networks evolve over time, they become denser, suffering more from readability and clustering issues. For example, in a network of twenty years of coauthorship for a specific conference, we observed two common behaviors that particularly challenge the network clustered representation: researchers moving across labs, and researchers co-supervising students. As we will discuss, duplications can provide a clearer picture of such relationships, but we must carefully consider subtle duplication variations and different visual designs.

3.1 When to duplicate?

As researchers move to new research labs they change coauthorship communities and become bridges between them. For example, George Robertson worked at Xerox PARC before moving to Microsoft Research. Should he be placed in PARC, MSR, or outside both of them? Considering ten or twenty years of collaboration, many researchers moved or collaborated with different labs (often more than 3), resulting in cluttered network representations that may be also suffering from arbitrary clustering. By duplicating these central actors, we can provide accurate views of the communities themselves (such as their size), while reducing the number of links displayed.

Another common problem is the representation of two senior researchers (from different research groups) supervising a group of common students. Should the students be placed in one group or the other, or should the two groups be merged? Here again we can use duplication in two ways: either by duplicating both senior researchers (to place each one in the community of the other), or by duplicating the group of students. In the second case, more actors are duplicated but it might be more relevant to show all students of each researcher.

Although these examples discuss coauthorship networks, duplication can be used in any type of social network where community assignment and clustering is ambiguous.

3.2 How to duplicate?

We investigated two different types of duplication for graph representations having different impacts on readability (Figure 3):

Clone: An exact copy of the node and all its connections is created, increasing the number of links in the graph and potentially causing clutter.

Split: The node is copied, but its connections are split between the original and the duplicated nodes. A visual connection between the original node and its duplicates is provided (directed link), but not between duplicates.

3.3 How to visualize duplications?

Understanding how central actors connect communities is important in social network analysis [31], and thus the visual connection between a node and its duplicates should be clear and easily accessible.

Existing approaches color-code all duplicated nodes and use their labels as disambiguation mechanisms [12], or interactively highlight the duplicates of a node when selected [23]. Other approaches link duplicates [1] to provide more immediate visual connections. Our hypothesis being that visual links are helpful, we examined links that would be easily distinguishable from regular graph links, while minimizing interference. Different preattentive features [29] were considered as parameters in the design of duplication links. To minimize interference with the perception of regular links and create a subtle effect, we rejected early on visualizations attracting attention such as animated links [4, 5], or curly and zig-zag type links such as the genogram of GenoPro [1]. Social networks contain a fair amount of regular links that often cross, thus dotted or dashed lines, or using angular changes or curvature in duplication links, would not be very effective as differentiation mechanisms.

We thus decided on using combinations of color hue, saturation and width, key visual variables to help differentiate duplication links at a glance [6, 7], while creating a subtle effect that can be ignored when users are not interested in the duplication. Two designs were considered: representing duplication links as thick de-saturated lines (*linkWidth*) and regular width links of different color (*linkColor*). For the *linkWidth* lines we chose grey color for a subtle effect, but grey was hard to see in the thinner *linkColor* and was replaced with light orange (same as actor nodes).

In the case of clone duplication all duplicates of a node are exactly the same, but for split duplication, the original node and its duplicates differ, an aspect we highlight in our links. Duplication links in split nodes thus fade out from the original towards the copy. To minimize confusion, we feel that for split nodes all duplicates should be derived from a single original node (1 level duplication) and not from other duplicates, easing the identification of the original node.

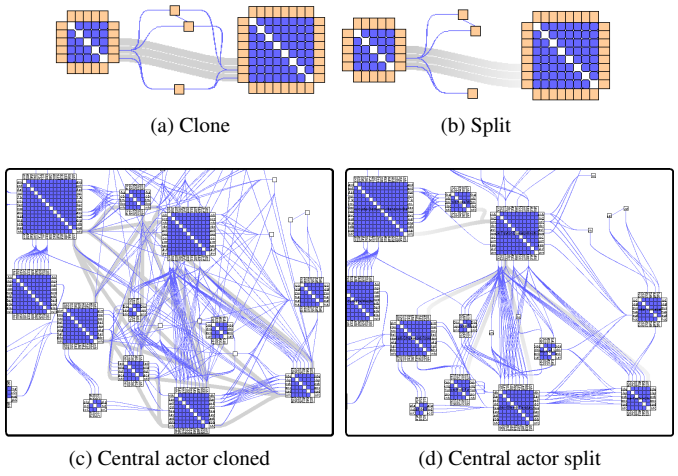


Fig. 3: Clone and Split duplications examples. (3c) and (3d) represent the same sub-set of a coauthorship network, a single central actor being duplicated in both cases.

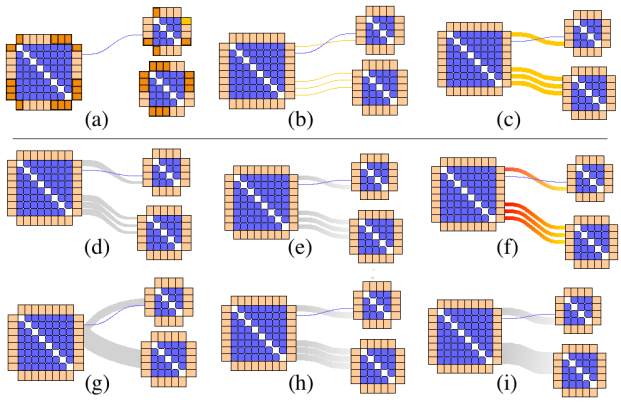


Fig. 4: Visual design alternatives for representing duplications links. Top row: clone duplication using (a) node color, (b) link color and (c) link color and large width. Second and third row: split duplication. As the direction of the duplication is important, we use variations in link (d) thickness, (e) saturation and (f) color. Duplications towards the same community can be grouped or "bundled" (g) linking matrices center or (h,i) by increasing links width.

4 CONTROLLED EXPERIMENT

An experiment was performed to determine the effect of the different duplications types and designs described above. Two types of duplication were used: complete *Clone* duplication (actor and all its connections duplicated) and *Split* duplication (only actor node duplicated). For both types, two design variations were considered for expressing the duplication connection (link between duplicates and original actor): links of the same width as other graph links, but of different *Color* (orange); and links of larger *Width* and faint grey color. Thus the examined duplication visualizations were: *cloneLinkColor*, *cloneLinkWidth*, *splitLinkColor* and *splitLinkWidth*. These were compared against two base case visualizations: first, clone duplication with color coding (orange) of duplicated nodes [12, 23], indicating their special nature *cloneNode*. No link between original actor and duplicates was present. And second, against *noDuplication* to explore potential issues in using any type of duplication.

During a pilot study, we found that the representation of duplication links using color (*cloneLinkColor* and *splitLinkColor*) performed somewhat worse than the visualization that represents duplication links as faint grey thick links (*cloneLinkWidth* and *splitLinkWidth*). This was especially true in tasks where duplication may hinder performance. As one participant noted "it is easier to perform tasks using the grey thick lines, because I can ignore them when I want to", indicating that grey links of larger width are easier to distinguish from other graph links and can be ignored more effectively. For our experiment we decided to use the grey thick links to visualize connections between duplicates, to limit interference with the remaining links of the graph.

Thus in the main study participants performed tasks using four visualizations: *cloneLink* and *splitLink* (links between duplicates are faint grey lines of large *Width* - Figure 3), *cloneNode* (no links between duplicates of actors - Figure 4a) and *noDuplication*. In all duplicated conditions (including *cloneNode*) participants could interactively select a duplicated node/actor to highlight all its duplicates in the graph in red [23].

4.1 Tasks

We decided to first evaluate duplications against two commonly used *graph readability* tasks [27, 13, 30]. We selected an overview task (RO) and a low-level detailed task (RD).

Task 1 (RO): *actorEstimation*
Participants were asked to compare the size (number of actors) of two sub-networks. To encourage estimation over accuracy, participants selected one of three answers: larger, equivalent or smaller.

Task 2 (RD): *actorConnectivity*
Participants were asked to enter the (topological) shortest distance between two actors as a numerical value.

In *social networks analysis*, it is important to identify community structures and their connections (C), as well as central actors and their influence on different communities (CA). To that end we selected another four commonly performed tasks [31]:

Task 3 (C): *communityConnectivity*
Participants were asked to select the two communities that share the larger number of central actors (strongest cohesion).

Task 4 (C): *communityCentrality*
Participants were asked to select the most central community, the community that shares central actors with the largest number of other communities.

Task 5 (CA): *articulationPoint*
Participants identified an actor that lies between a community and the rest of the network. Articulation point are described as actors that, when removed, cause one or more communities to become disconnected from the network (graph cut-point).

Task 6 (CA): *mostConnected*
Participants were asked to identify and select the most connected actor: the actor with most connections to other actors (equivalently the node with the largest degree).

These six tasks are representative of general purpose graph readability and social network practices. Moreover, they were selected so as to cover analysis tasks ranging in granularity from overview understanding (*actorEstimation*), intermediate community structure (*communityConnectivity, communityCentrality*) and central actors (*articulationPoint, mostConnected*), to detailed tasks focusing on specific nodes or links (*mostConnected, actorConnectivity*).

Since duplication affects the general layout of the network, the selected tasks are relevant to the topology of the network, and not specific attributes of individual actors or links (another set of common social network analysis tasks [27]).

4.2 Dataset

Generating synthetic representative graphs is still a challenge. As explained in [16], current small-world generators do not provide realistic models. Therefore, to conduct the experiment using realistic graphs that follow properties of small-world networks, we used subsets of actual datasets: nearly complete coauthorship data for twenty-five years of ACM CHI (Human Factors in Computing Systems) conferences and twenty years of ACM UIST (User Interface Software and Technology) conferences. The labels of actors in the dataset were replaced by codes to avoid interpretation issues. The subsets both contained 130 actors (out of 300 from UIST and 1500 from CHI) in order to fit the graphs on a single screen, suppressing any navigation issues and confounding factors. Subsets of the graphs were carefully chosen (but not altered to preserve the small-world properties) to provide a balanced design: we balanced the link density between graphs, the number of communities, the number of duplicated actors and ensured that communities of actors are always cliques (to make their identification in the *noDuplication* visualization easier).

4.2.1 Communities and duplications

In social networks, it is important to see both communities and central actors but existing automatic clusterings only give communities. For the experiment purposes, we ensured communities were all cliques and assumed that actors falling into two or more cliques (connected to all actors or the communities) were central actors. We used the edge-betweeness clustering algorithm [14] implemented in the JUNG [26] package, and edited a few of the communities a posteriori to ensure they were all cliques. All actors belonging to two or more cliques were considered central actors and duplicated. In the splitLink, original actors are placed in the largest community. Similarly, in the noDuplication condition, central actors are placed in the largest community they belong to. We used NodeTrix [17] to represent the topological structure of the social network. The graph layout was performed using a manual layout minimizing edge-crossing, tuned from an initial LinLog [25] algorithm layout.

4.2.2 Density

In order to better understand the effect of duplication, participants were asked to perform the mentioned tasks in graphs of two types of density (*Sparse* and *Dense*). We selected graph density as this attribute has been proven to affect task performance in graph understanding [13]. The density (ratio of existing number of links over all possible links in a graph, a ratio that is fairly small in small world networks [32]) was 0.14 for the sparse, as opposed to 0.19 for the dense one. As both networks had small-world properties, they had a high clustering coefficient (0.86 in both cases).

4.3 Participants and apparatus

Twelve participants (1 female) took part in the study. Aged from 23 to 40, they were all researchers or students of a graph drawing research group, to ensure familiarity with computers and graph representations.

We used a 3GHz Pentium IV computer with 1GB of RAM and one 19" screen during the controlled experiment. Participants entered their answers using mouse or keyboard.

4.4 Experimental design

A repeated measures within-participant full factorial design was used. The independent variables were *Task* (actorEstimation, actorConnectivity, communityConnectivity, communityCentrality, articulationPoint, mostConnected), visualization *Vis* (noDuplication, cloneNode, cloneLink, splitLink), and density *Dens* (sparse, dense).

Participants were randomly assigned to 4 groups of 3. In each group participants used all 4 visualizations, in an ordering balanced using a Latin square. For each visualization participants completed a single block, containing 2 trial repetitions for all combinations of task (6) and density (2). To reduce memorization of graph layouts between trials, graph labels were randomly generated and the entire graph layout was rotated randomly. The experiment consisted of:

4 visualizations x **2** densities x **6** tasks x **2** repetitions x **12** participants = **1152** trials

Before the experiment, participants were interviewed to gather information about their previous experience with graphs and visual representations. A tutorial sheet introduced the NodeTrix representation, the duplications designs and each of the six tasks to complete. An experimenter was present to answer all questions. Participants could then practice with the experimental system for random trials on a training dataset. Training was also given at the beginning of each visualization block. The training sessions lasted 10 minutes on average and ended when the experimenter ensured all tasks and visualizations were understood. At the end of the experiment, participants completed a questionnaire eliciting their visualization preference per task and overall, and commented on the use of the visualizations.

Participants were asked to perform the task correctly as fast as possible. To prevent random answers, if participants felt unable to answer a question, they were allowed to skip it. To limit the experiment duration, task completion time was limited to 60 seconds. Neither of these events occured.

5 Hypotheses and Results

5.1 Task 1: actorEstimation (RO)

H: The use of duplication will degrade the accuracy and performance time for comparing the size of two graphs, as duplicated actors result in larger number of nodes in a graph.

Success Rate (SR): Surprisingly, the Friedman's chi square test showed no significant effect of *Vis*. This task was very error prone for all visualizations: cloneNode (33% success rate), splitLink (45%), cloneLink (50%) and noDuplication (56%).

Performance Time (PT): A significant effect of *Vis* on time was present ($F_{3,33} = 7.43$, $p < .05$), as well as a significant *Vis* x *Dens* interaction ($F_{3,33} = 9.94$, $p < .0001$). Mean times were: splitLink (12.17sec), no duplication (14.18sec), cloneNode (15.16sec) and cloneLink (17.01sec). Post-hoc pairwise mean comparisons showed cloneLink to be significantly slower than splitLink but all other pairs were not statistically significant (all adjustments Bonferonni). Contrary to our prediction, duplications did not degrade performance time and accuracy for this task.

User Preference (UP): Not surprisingly 9 of 12 participants preferred noDuplication for this task, explaining that "the networks to compare feel more different when there are duplications". The remaining preferred either cloneLink or splitLink. However several reported that they "usually overcompensate the number of duplications" and had low confidence in their answers. 7 out of 12 thus ranked this task as the most difficult.

5.2 Task 2: actorConnectivity (RD)

H: The introduction of duplication will negatively affect the performance time and accuracy of counting the distance between two actors, as extra duplication links are introduced between actors.

SR: The Friedman's test showed a significant effect of *Vis* ($p < .05$) on Success Rate. Pairwise comparison using the Wilcoxon's test showed that noDuplication (69% success rate) was more accurate than cloneNode (53%) and cloneLink (55%), but not splitLink (65%). For this unique task, we avoid memorization effects by using two different distances to count for each trial: 3 (D3) and 4 (D4). Thus we varied the tasks difficulty and added this independant variable to our analysis. The Wilcoxon's test reveals a significant difference between the task difficulty: D4 (25% success rate) is far more error prone than D3 (95%). If we split the results by task difficulty, the Friedman's test reveal a significant difference for the most difficult task D4 ($p < .05$). The Wilcoxon's test shows that noDuplication and splitLink perform both better than cloneLink (17% success rate). No significant difference is revealed betwen noDuplication (42%) and splitLink (33%).

PT: A significant effect of *Vis* on Time was present ($F_{3,33} = 5.99$, $p < .05$), as well as a significant *Vis* x *Dens* interaction ($F_{3,33} = 10.31$, $p < .0001$). Mean times were: noDuplication (21.13sec), splitLink (23.96sec), cloneLink (24.90sec) and cloneNode (27.66sec). Post-hoc pairwise mean comparisons showed cloneNode to be significantly slower than both cloneLink and noDuplication. Analysis split by task difficulty showed that there was no difference in *Vis* in the sparse graphs. In the dense graphs cloneNode was significantly slower than all other visualizations, whereas noDuplication was faster than cloneNode and cloneLink, but not from splitLink (all adjustments Bonferonni). Although we expected duplications to degrade time performance, this was only true in the clone duplication cases (cloneNode and cloneLink). SplitLink did not significantly degrade the performance time or success rate compared to noDuplication.

UP: Surprisingly only 4 out of 12 participants preferred the noDuplication condition for counting the distance between two actors, saying it was easier "as you did not need to go through an extra link of cost 0". From the duplication conditions, splitLink was generally preferred (4/12 ranked it first, 5/12 second). Several participants reported that it could be "tricky to see where the link goes" in noDuplication whereas "duplications cleans the graph" and "makes it easier to see the shortest path".

5.3 Task 3: communityConnectivity (C)

H: The introduction of duplications (especially using links) will allow for faster and more accurate identification of the two communities that share the larger number of actors.

SR: The Friedman's test showed a significant effect of *Vis* ($p < .0001$). As expected, the Wilcoxon's test showed that all three duplication visualizations performed better than noDuplication (only 33% sucess rate). Identifying actors (nodes) that are clustered in one community but belong to others is a harder task than counting duplicated actors. SplitLink was the most accurate visualization (96% success rate), followed by cloneNode (86%) and cloneLink (82%). splitLink was also significantly more accurate than cloneLink ($p < .05$). The better accuracy of splitLink is due to the "cleaner" resulting graphs, where links of the duplicated actors are not duplicated themselves.

PT: There was a significant main effect of *Vis* on *Time* ($F_{3,33} = 22.63$, $p < .0001$) and a significant *Vis* x *Dens* interaction effect ($F_{3,33} = 6.5$, $p < .0001$). Mean times were: splitLink (20.28sec), cloneLink (20.72sec), cloneNode (31.07sec) and noDuplication (35.58sec). Post-hoc mean comparisons showed splitLink and cloneLink to be significantly faster than both cloneNode and noDuplication. This behavior was present in sparse graphs, but in the dense graphs noDuplication was significantly slower than all the duplication techniques. Indeed times in the case of duplication using links (splitLink and cloneLink) were generally faster, as it is easier to notice the width of the duplication link bundle going to different communities to identify the two communities that share the most actors. Nevertheless, in dense graphs even simple cloneNode outperformed noDuplication, as looking for duplicated actors is easier than counting links from actors to other communities.

UP: 11 of 12 participants preferred duplication techniques for this task, with splitLink and cloneLink being the most preferred (5/12 each). Participants reported that "grey lines tell the story", that it was easy to estimate the width of the grey lines to see communityConnectivity. They described splitLink as "cleaner", but as grey lines between multiple duplicates of the same actor were missing, they their confidence was lower. No one preferred noDuplication.

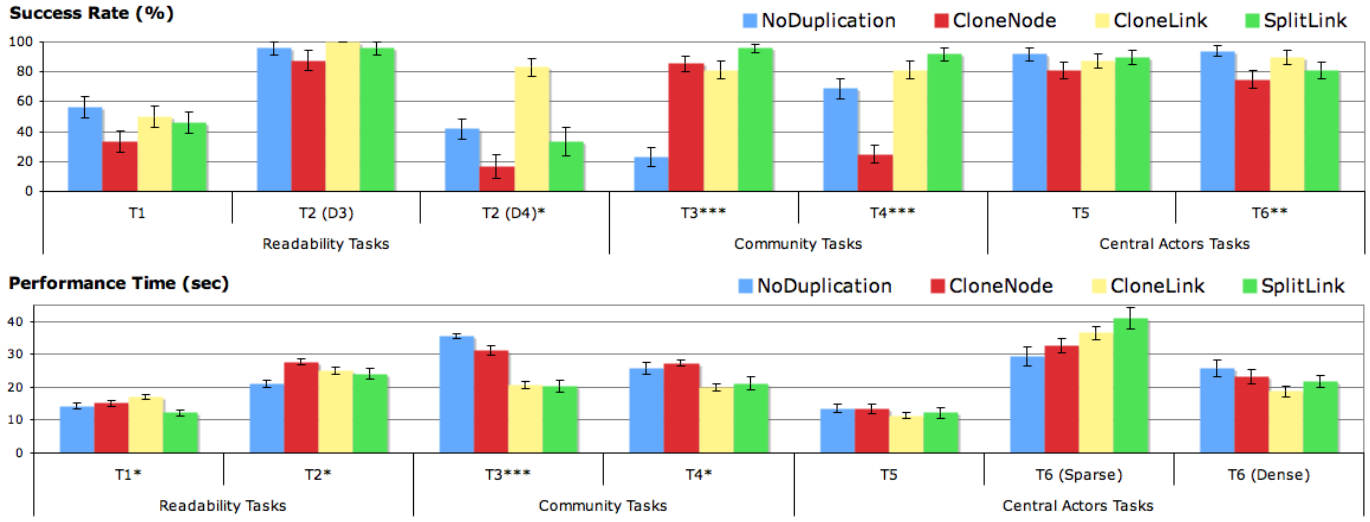


Fig. 5: Success rate and time per task per visualization. Significant differences are indicated by * ($p < .05$), ** ($p < .001$) and *** ($p < .0001$).

5.4 Task 4: communityCentrality (C)

H: Duplications (especially using links) will allow for faster and more accurate identification of the most central community, that shares actors with the most other communities.

SR: Friedman's test showed a significant effect of *Vis* ($p < .0001$). Wilcoxon's test showed cloneNode, (25% sucess rate) as more error-prone than the remaining techniques. Also splitLink (92%) performed significantly better ($p < .05$) than noDuplication (69%), with no difference between cloneLink (82%) and noDuplication (69%).

PT: There was a significant main effect of *Vis* on *Time* ($F_{3,33} = 3.07$, $p < .05$) and a significant $Vis \times Densinteractioneffect (F_{3,33} = 6.78$, $p < .05$). Mean times were: cloneLink (20.01sec), splitLink (21.13sec), noDuplication (25.83sec) and cloneNode (27.26sec). Post-hoc pairwise mean comparisons showed cloneLink to be significantly faster than cloneNode in all graph densities. All other pairs were not significantly different. Contrary to our expectations, the duplication conditions using links did not yield significantly faster times (although their mean times were faster overall), indicating that the normal links between shared actors in the noDuplication condition are enough to identify central communities. Nevertheless, duplications using links make this identification more accurate.

UP: Not surprisingly 11 of 12 participants preferred duplication techniques for this task, with cloneLink and splitLink being the most preferred (often ranked at the same position)(6/12 for each). A participant commented the grey links helped a lot as "you can stop paying attention to the blue (regular) lines and concentrate on the grey only". Several participants reported that splitLink could be misleading as they thought the choice of the original node was decisive. Almost all of them reported that the cloneNode "required exploration" and "many clicks and memorization".

5.5 Task 5: articulationPoint (CA)

H: Duplications using links will help in the identification of articulation points (actors bridging two communities), as fewer links are present between communities.

SR: Friedman's test showed no significant effect of *Vis* on Success Rate: noDuplication(92% success rate), splitLink (90%), cloneLink (88%) and cloneNode (82%).

PT: There was no significant effect of *Vis* on time, with mean times for duplication techniques being slightly faster: cloneLink (11.32sec), splitLink (12.18sec), cloneNode (13.53sec) and noDuplication (13.85sec).

UP: Almost all participants considered the representations equivalent for this task. 8 of 12 participants reported that this task was the easiest. When asked about their strategy, almost all replied that they "look at the network periphery to find a community linked by a few or single connection". Several reported that splitLink was confusing as they "wondered about missing links".

5.6 Task 6: mostConnected (CA)

H: The introduction of duplications will make the identification of the most connected actor (larger number of connections) harder, as the actors are now in multiple communities.

SR: Contrary to previous studies [13], where datasets were artificially generated and the most connected actor degree increased of 5% between trials, we did not modify our datasets. In the sparse graph, two actors were candidates with only a small degree difference between them, therefore they were both considered as a right answer. Friedman's test showed a significant effect of *Vis* ($p < .01$). The Wilcoxon's test revealed a significant difference between noDuplication(94% success rate) and cloneNode(75%) but no difference between noDuplication and cloneLink(90%) or splitLink(81%). CloneLink performed significantly better than cloneNode and splitLink($p < .05$).

PT: There was no significant main effect of *Vis* on *Time* ($F_{3,33} < 1$), but a significant main effect of *Dens* ($F_{3,33} = 46.21$, $p < .0001$) and a significant *Vis* x *Dens* interaction effect ($F_{3,33} = 6.49$, $p < .001$). Mean times were: cloneNode (27.88sec), noDuplication (27.55sec), cloneLink (27.62sec) and splitLink (31.46sec). Post-hoc pairwise mean comparisons showed splitLink and duplication cases tend to be slower than noDuplication in the sparse graph, likely due to the ambiguous couple of candidates as it does not occur in the dense one.

UP: 8 of 12 participants preferred duplication techniques for this task. They explained that they roughly estimated the most connected actor as one that is part of many communities. They also reported that splitLink looks cleaner but that the missing links greatly lower their confidence. Half of the participants reported this task as most difficult.

5.7 Overall user preferences and comments

Almost all participants reported that cloneNode, used often in practice, would be useless without node highlighting. Even with it, they describe their strategy as "a trial and error process", tedious and cognitively demanding. Surprisingly, 6 out of 12 participants counted splitLink as their preferred visualization overall (2 for noDuplication, 1 for cloneNode and 3 for cloneLink). When asked why (as splitLink was not ranked first in most tasks), roughly all participants reported that "it looks cleaner" but they feel less confident, as links between multiple duplicates are not present (only between the original and duplicates). However, they all commented that this feeling would probably disappear with more pratice.

6 DISCUSSION

Our experiment showed that the most common technique for duplications (cloneNode) performed very poorly, even when coupled with interactive feedback, and that visual duplication links were more effective. Our design choices attempted to create two types of links easily distinguishable by using a subtle combination of color, intensity and width. We hoped that users could pre-attentively identify and ignore the duplication links when needed, thus not affecting the performance of readability tasks (RD and RO) and similarly ignore the regular links while performing community tasks(C). The good performance of duplication across tasks and qualitative results of our experiment indicate that this was achieved. Indeed most of our participants clearly stated that they could ignore one or the other type of links, which was helpful when performing the tasks.

6.1 Readability tasks (RO, RD)

We expected duplications to degrade the success rate and performance time of overview and detailed readability tasks, but results from our experiment proved that *split duplication was as effective as noDuplication for both tasks*.

As our overview task (RO) was particularly error prone, results showing no significant impact of duplications on the performance time and success rate must be interpreted carefully. Previous studies [13] also showed that counting nodes or links is a particularly error prone task. Some participants reported that the visualization was not familiar and thus estimating the number of nodes was hard. During the pilot, one participant noted that "the matrices increased the visual effect of the number of nodes (because you compare areas larger than the actual number of actors they contain)". This could be considered an artefact of the NodeTrix representation, but should have affected noDuplication and duplication cases similarly.

Interestingly, duplications do not strongly affect the detailed task(RD). Participants even reported the task was easier to perform with splitLink as it suppressed a number of link crossing, making it easier to follow the connections. For clone duplications, the detailed task seems more affected in denser networks, where clone duplication results in multiple copies of regular connections for all duplicates.

6.2 Community tasks (C)

Our experiment showed that *duplications helps identifying the connections between communities*. The visual design showing links has been proven more efficient and almost all participants reported the grey lines were really helpful. Qualitative results from the pilot showed that using a grey thick line instead of a thin line of a different color provides two levels of readability. All participants of the pilot reported that it was easier to concentrate on grey lines and ignore other types of links (and vice versa).

The introduction of *duplications also helps to find more accurately a central community*, especially using splitLink. Here it is easier to follow the direction of the different duplication links (leaving the most central community), than in cloneLink or noDuplication which suffer from regular link clutter. CloneNode performs really poorly, as the only way to identify central communities is to select all the duplicated actors of a community and identify (through highlighting) the other communities it is connected to.

As duplications solve the problem of ambiguous clustering, they *display the true size of communities*, providing more accurate comparisons between communities. This task was not part of our experiment, but part of our motivation.

6.3 Central actors tasks (CA)

The concept of central actors in a social network is intuitive but hard to perform in a controlled experiment. In practice, it requires several measures (such as computing several centrality metrics) and the interpretation of the actors' attributes. For our experiment, we selected two common tasks that are simple to explain and validate: finding an articulation point and identifying the most connected actor. We expected that these tasks would be positively and negatively affected respectively by duplications. Our predictions were wrong, as *no difference was shown for the articulation point* while *duplications were preferred for the most connected actor*.

While using duplications, we expected that summing up all duplicates and their links to identify the most connected actor, would degrade the overall performance, but it only did for the ambiguous case (Figure 5, T6 (sparse)). Most participants reported that they considered the most connected actor as one part of many communities. Duplications were thus preferred, as participants counted the outgoing grey lines or the highlighted duplicates of an actor. This is the only task where cloneNode performed well, as participants directly clicked only duplicated nodes (that are of a discrete color) to find one disseminated across many communities, instead of trying to identify the directions of the grey links.

Although we expected a positive effect of duplications on the articulation point identification, no difference between visualizations was found. All participants stated that they looked at the periphery of the network to find an isolated community, searching for a single grey or blue line connecting it to the rest of the network. This strategy yields good performance in an experiment framework (where participants are required to be fast), but results are to be interpreted carefully. In practice, the most interesting articulation points are the ones which disconnect larges parts of a graph and are usually placed in the center of the graph (where there is far more clutter than on the periphery). As this is a lengthy task to perform (finding an articulation point that disconnects the graph in two maximal subgraphs) and far more cognitively demanding (comparing the impact of several articulation points), we did not include it in our controlled experiment. Thus our results may not reflect the real impact of duplications on the identification of "important" articulation points. Further experimentation is needed to validate our hypothesis that duplications reduce visual complexity, making important articulation points easier to find.

6.4 Duplication guidelines

Based on our experimentation, we propose the following general guidelines for node duplications in clustered graph representations:

When to duplicate?
(1) To reduce visual complexity in graphs that have many actors shared among communities.
(2) To emphasize central actors that connect multiple communities. To highlight their importance, such actors may also be extracted from their communities when duplicated.
(3) To provide accurate community-centered views, an important aspect of many social network analysis tasks.

How to duplicate?
(4) Using either split or clone, but not a combination of the two, as they are both complex representations.
(5) Clone can be used as base case, as it requires less practice (at the expense of cluttering the network).
(6) Split reduces visual complexity, but interactive highlighting of the duplication links may be required for novice users.

How to visualize duplication?
(7) Simple colored nodes are not enough for representing duplications. Links between duplicates are more effective.
(8) To increase readability, visual links that connect duplicates should be easily distinguishable from other graph links.
(9) Interactive highlighting of duplicated nodes and links is desirable.

6.5 Applicability to node-link diagrams

Our results apply to NodeTrix representations and more generally to clustered node-link diagrams. The difference between these representations lies in how communities are visualized but both suffer from the same readability and ambiguous clustering problems. While NodeTrix improves intra-community readability by removing links crossings, both representations could benefit from duplications to improve inter-community readability (by reducing the number of links) and to provide more accurate views of the communities, showing their actual size and highlighting shared actors.

Concerning standard node-link diagrams (without concrete circled clusters), node duplication requires further investigation as its impact directly depends on the graph layout. Eades and Mendoca presents an early example of duplication to reduce edge crossing [9]. In the context of social network analysis, node duplication would be particularly suited for layouts making clusters emerge. For instance, spring layout algorithms make clusters of nodes appear, shared actors being naturally placed between them. This may cause clutter as these actors have many links towards the communities they belong to. Duplicating shared actors should impact the spring layout, potentially placing each duplicate in a community and thus reducing the inter-community clutter. However, further experimentation is required to understand how duplicating impact these layouts.

7 Conclusion and Future work

Social network visualization is becoming increasingly important with the creation of new online communities and the need to monitor and analyse their evolution and structure. Major challenges facing social network visualization and analysis include the lack of readability of the resulting large graphs, and the often ambiguous assignment of actors shared among multiple communities to a single community. In this paper we propose using actor duplication in social networks in order to assign actors to multiple communities without greatly affecting the readability. After investigating different design alternatives for representing duplications, we conducted a controlled experiment using 6 tasks relevant to both graph readability and social network analysis.

Our results are summarized as a set of design guidelines applicable to clustered node-link diagrams and the recent NodeTrix representation; and useful to both analysts trying to communicate theirs findings using graphs with duplications, as well as visualization experts attempting to create automatically generated layouts using duplication.

Our exploration showed actor duplication to be very promising in social network analysis and representation: community centered tasks were greatly benefited, whereas other tasks were affected little, if at all. Although these tasks were chosen to be representative of current practices, further analysis on a larger set of tasks needs to be performed. Furthermore, duplicating actors presenting other properties than being shared (highly connected or high betweeness centrality actors for instance) provides also interesting prospects.

In the future, we plan to investigate how interaction can support the creation, edition and visualization of node duplications. Our participants already suggested a couple of interesting directions: providing interactive feedback on links, by mousing over them and highlighting all links of the duplicated actor; showing several levels of duplication detail, with a default state in which duplications links are grouped in a bundle between communities (estimation), but become independent (countable) on mouse over; and finally a smooth animation to merge back all duplicates of a node into a single one, to improve the understanding of duplications effects and the impacted network areas.

Acknowledgements

This work was supported in part by the joint Microsoft-Research/INRIA ReActivity Project.

References

[1] Genopro. http://www.genopro.com/.

[2] J. Abello, F. van Ham, and N. Krishnan. Ask-graphview: A large scale graph visualization system. *IEEE TVCG journal*, 12(5):669–676, 2006.

[3] D. Auber, Y. Chiricota, F. Jourdan, and G. Melançon. Multiscale visualization of small world networks. In *Proc. of IEEE Symposium on Information Visualization (InfoVis'03)*, pages 75–81. IEEE Press, 2003.

[4] L. Bartram and C. Ware. Filtering and brushing with motion. *Information Visualization*, 1(1):66–79, 2002.

[5] L. Bartram, C. Ware, and T. Calvert. Moticons: detection, distraction and task. *Int. J. Hum.-Comput. Stud.*, 58(5):515–545, 2003.

[6] J. Bertin. *Semiology of graphics*. University of Wisconsin Press, 1983.

[7] S. K. Card and J. Mackinlay. The structure of the information visualization design space. In *Proc. of IEEE Symposium on Information Visualization (InfoVis'97)*, Washington, DC, USA, 1997.

[8] G. Di Battista, P. Eades, R. Tamassia, and I. G. Tollis. *Graph Drawing: Algorithms for the Visualization of Graphs*. Prentice Hall PTR, 1998.

[9] P. Eades and C. F. X. de Mendonca N. Vertex splitting and tension-free layout. *Proc. of Graph Drawing'95, LNCS*, 1027:202–211, 1995.

[10] P. Eades and Q.-W. Feng. Multilevel visualization of clustered graphs. In *Proc. Graph Drawing, GD*, number 1190, pages 101–112, Berlin, Germany, 18–20 1996. Springer-Verlag.

[11] P. Eades, Q.-W. Feng, and X. Lin. Straight-line drawing algorithms for hierarchical graphs and clustered graphs. In *Proc. of the Symposium on Graph Drawing'96*, pages 113–128, London, UK, 1997. Springer-Verlag.

[12] P. Eklund, N. Roberts, and S. Green. Ontorama: Browsing rdf ontologies using a hyperbolic-style browser. In *Proc. of the First International Symposium on Cyber Worlds (CW'02)*, page 0405, Washington, DC, USA, 2002. IEEE Computer Society.

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

[14] M. Girvan and M. E. Newman. Community structure in social and biological networks. *Proc Natl Acad Sci U S A*, 99(12):7821–7826, June 2002.

[15] N. Henry and J.-D. Fekete. MatrixExplorer: a Dual-Representation System to Explore Social Networks. *IEEE TVCG journal (IEEE InfoVis'06 proc.)*, 12(5):677–684, 2006.

[16] N. Henry and J.-D. Fekete. Matlink: Enhanced matrix visualization for analyzing social networks. *Proc. of the 11th IFIP TC13 International Conference Interact'07*, LNCS 4663:288–302, 2007.

[17] N. Henry, J.-D. Fekete, and M. J. McGuffin. NodeTrix: a hybrid visualization of social networks. *IEEE TVCG journal (IEEE InfoVis'07 Proc.)*, 13(6):1302–1309, 2007.

[18] I. Herman, G. Melançon, and M. S. Marshall. Graph visualization and navigation in information visualization: A survey. *IEEE Transactions on Visualization and Computer Graphics*, 6(1):24–43, 2000.

[19] D. Holten. Hierarchical edge bundles: Visualizaiton of adjacency relations in hierarchical data. *IEEE Transactions on Visualization and Computer Graphics*, 12(5):741–748, 2006.

[20] A. K. Jain, M. N. Murty, and P. J. Flynn. Data clustering: a review. *ACM Computing Surveys*, 31(3):264–323, 1999.

[21] H. Kang, C. Plaisant, B. Lee, and B. B. Bederson. Netlens: Iterative exploration of content-actor network data. *Proc. of IEEE Symposium on VAST*, pages 91–98, 2006.

[22] B. Lee, M. Czerwinski, G. Robertson, and B. B. Bederson. Understanding eight years of infovis conferences using PaperLens. In *Proc. of the IEEE Symposium on Information Visualization*, page 216.3, Washington, DC, USA, 2004. IEEE Computer Society.

[23] B. Lee, C. S. Parr, C. Plaisant, B. B. Bederson, V. D. Veksler, W. D. Gray, and C. Kotfila. Treeplus: Interactive exploration of networks with enhanced tree layouts. *IEEE TVCG journal*, 12(6):1414–1426, 2006.

[24] M. McGrath, J. Blythe, and D. Krackhardt. The effect of spatial arrangement on judgments and errors in interpreting graphs. *Social Networks*, 19(3):223–242, 1997.

[25] A. Noack. Energy-based clustering of graphs with nonuniform degrees. In P. Healy and N. S. Nikolov, editors, *Proc. of the Graph Drawing (GD'05)*, pages 309–320, Limerick, Ireland, 2005. Springer-Verlag.

[26] J. O'Madadhain, D. Fisher, P. Smyth, S. White, and Y.-B. Boey. Analysis and visualization of network data using jung. *Journal of Statistical Software, Preprint*, pages 1–35, 2005.

[27] C. Plaisant, B. Lee, C. S. Parr, J.-D. Fekete, and N. Henry. Task taxonomy for graph visualization. In *BEyond time and errors: novel evaLuation methods for Information Visualization (BELIV'06)*, pages 82–86, Venice, Italy, 2006. ACM Press.

[28] H. Purchase. Which aesthetic has the greatest effect on human understanding. In G. Di Battista, editor, *Graph Drawing, Rome, Italy, September 18-20, 1994*, pages pp. 248–261. Springer, 1998.

[29] A. Treisman. Preattentive processing in vision. *Computer Vision, Graphics, and Image Processing*, 31(2):156–177, 1985.

[30] C. Ware, H. Purchase, L. Colpoys, and M. McGill. Cognitive measurements of graph aesthetics. *Information Visualization*, 1(2):103–110, 2002.

[31] S. Wasserman and K. Faust. *Social Network Analysis*. Cambridge University Press, 1994.

[32] D. J. Watts and S. H. Strogatz. Collective dynamics of 'small-world' networks. *Nature*, 393:440 – 442, 1998.