An Extended Evaluation of the Readability of Tapered, Animated, and Textured Directed-Edge Representations in Node-Link Graphs

Danny Holten* Eindhoven University of Technology
Petra Isenberg† INRIA
Jarke J. van Wijk‡ Eindhoven University of Technology
Jean-Daniel Fekete§ INRIA

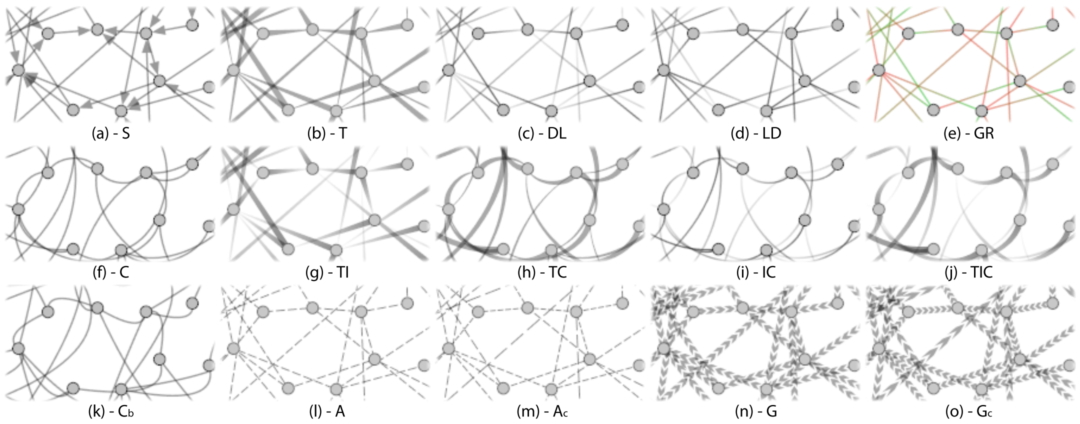

Figure 1: All directed-edge representations used in our initial (a to j), follow-up (b, k, l), and current study (b, l, m, n, o). (a) standard arrow – S, (b) tapered – T, (c) dark-to-light – DL (a.k.a intensity – I), (d) light-to-dark – LD, (e) green-to-red – GR, (f) curvature – C, (g) tapered-intensity – TI, (h) tapered-curvature – TC, (i) intensity-curvature – IC, (j) tapered-intensity-curvature – TIC, (k) biased curvature – C_b, (l) animated – A, (m) animated compressed – A_c, (n) glyph – G, and (o) glyph compressed – G_c.

Abstract

We present the results of a study comparing five directed-edge representations for use in 2D, screen-based node-link diagrams. The goal of this work is to extend the understanding of tradeoffs and best practices for the representation of edges in directed graphs and to help practitioners choose among different options. Our work applies to graphs in which directed links are depicted using lines connecting the nodes. We tested five different edge representations chosen carefully based on user feedback to thoroughly cover the directed-edge design space. We also investigated how the use of pattern compression affects performance and subjective user preference. The article presents detailed results regarding the significant performance and preference differences between directed-edge representations and provides practical recommendations on their use.

Index Terms: H.5.2 [Information Interfaces and Presentation]: User Interfaces—Evaluation/Methodology I.3.3 [Computer Graphics]: Picture/Image Generation—Line and Curve Generation H.1.2 [Models and Principles]: User/Machine Systems—Human Information Processing

1 Introduction

Graph-based data has become ubiquitous these days, not least due to the increasing popularity of social networking sites on the internet. Graphs represent a collection of elements – called vertices or nodes – as well as the connections between these elements – called edges or links. Apart from social network graphs, where vertices represent individuals and edges represent acquaintance, the following lists some additional examples of common graph-based data:

- *Traffic networks*, where vertices and edges are used to represent locations and traffic routes, respectively;
- *Computer networks*, where vertices and edges are used to represent PCs and network connections, respectively;
- *Scientific citation networks*, where vertices and edges are used to represent papers and citations between papers, respectively.

In graphs, edges often have an associated weight and direction. Edge weight might be used to indicate the strength, importance, or cost of an edge. Edge direction can be used to signify the directionality of what an edge represents, e.g., which paper holds a citation and which paper is being cited in case of a citation network graph.

An often-used and intuitive visual graph representation is the node-link diagram, in which vertices are generally represented as circular nodes and edges as straight or curved links (lines). Edge weight is commonly depicted by varying the width of a link, while edge direction is generally depicted using an arrow representation, i.e., a line with an arrowhead at the target node. However, our earlier work [15] showed that arrowheads are a poor choice for depicting directionality in all but very simple graphs. The main reason for poor performance was visual clutter due to arrowhead overlap, which significantly hinders effortless and correct observation of edge direction. Various approaches can be used to overcome this, such as user interaction, using a matrix instead of a node-link representation, improving the node-link layout, or improving the directed-edge representation. However, user interaction is not suited for statically depicted graphs such as printouts; matrix representations are less intuitive than node-link graphs [12, 17]; and layout improvement is generally tackled by *node*-placement optimization, while the arrowhead-clutter problem stems from a suboptimal *edge* representation. We were therefore motivated to focus our efforts on finding improved directed-edge representations.

The question arises as to which alternatives to the traditional arrow are the most suited for directed graphs and how the per-

*e-mail: d.h.r.holten@tue.nl
†e-mail: petra.isenberg@inria.fr
‡e-mail: vanwijk@win.tue.nl
§e-mail: jean-daniel.fekete@inria.fr

formance of different representations varies across different graph characteristics. In this paper we describe an extensive study in which we tested five different directed edge representations for varying graph densities and edge lengths. In particular, we were interested in testing whether edge representations would perform particularly well if they encoded not only directionality but also edge length. In [15], we found that a tapered representation (T; Figure 1b), i.e., one where width varies along the length of an edge from wide to narrow, was the only representation with a clear indication of edge length and it outperformed all other techniques. In this study, we explore other representations such as an animated and textured link which encode edge length through pattern compression using a 2D, screen-based testing environment and compare it with T. We conclude with implications for practitioners which highlight the tradeoffs of the tested techniques in terms of performance as well as their applicability in several design contexts.

The remainder of the paper is organized as follows. Section 3 describes previous work on directed-edge visualization as well as its evaluation. Section 4 gives an overview of graph generation and layout, the evaluated representations, and the selection of node pairs during trials. Experiment design and hypotheses are presented in Section 5, followed by Section 6, which presents the statistical analysis. Practical recommendations on directed-edge usage are provided in Section 7. Finally, Section 9 presents our conclusion and suggestions for future work.

2 Study Background

The study presented in this paper follows a stream of our recent research with the goal of providing a clear picture of the directed-edge design space and the tradeoffs when choosing edge presentations for node-link graphs. Inspired by our initial results from [15], we first conducted a smaller follow-up study which shed initial light on our current study problem. The smaller study has not been published apart from a technical report [14] so some of its findings are re-iterated to explain our choices, hypotheses, and conclusions.

Study 1: Our initial study [15] had participants perform tasks on a collection of graphs using different directed-edge representations to investigate which representation performed best in terms of reading time and correctness. A tapered representation (T) clearly outperformed the standard arrow (S), clockwise curvature (C), and color-/intensity-based representations, i.e., dark-to-light (DL), light-to-dark (LD), and green-to-red (GR). It also tested combinations of these representations: since participants showed a slight preference for DL, they chose this as the representative for the intensity (I) class and combined it with T and C, leading to TI, TC, IC, and TIC. Contrary to what they expected, stacking visual cues did not result in significant performance gains, hinting at possible interference instead of reinforcement effects. Figures 1a–1j show all representations from our initial study.

Study 2: Our smaller follow-up study [14] evaluated the performance of T, the best representation according to our initial study, against animation (A) and *biased* curvature (C_b), proposed by Fekete *et al.* [9] as a possible improvement to standard curvature (C) (Figures 1b, 1k, and 1l). In summary, T still outperformed C_b, regardless of the added bias, but A proved to be on par with T.

Current Study: In this paper, we describe the results of our most recent study, for which we chose the most promising candidates from our previous studies, i.e., T and A. We furthermore added a textured representation – "glyph" (G) – to cover the directed-edge design space further. C and C_b were dropped due to their poor performance in the previous experiments. The excellent performance of the edge-length indicating T representation motivated us to investigate how the use of edge-length indicating *pattern compression* in case of animated (A_c) and glyph (G_c) representations would affect performance (Figures 1b and 1l–1o). We furthermore evaluated how performance in terms of reading time and correctness are affected by three levels of graph density (sparse, medium density, and dense) as well as two link-length classes (short and long). Various subjective preference aspects were measured as well by means of a post-study questionnaire.

Although we aimed to cover visual modalities and cues for depicting link length (and direction) as well as possible, we note that due to feasibility constraints we only tested the most basic candidates for animation, compression, and texturing to first determine if such modalities can actually increase performance at all before investigating more advanced alternatives. A similar remark holds for the combination of modalities, different graph densities and layout, and experimental tasks: we chose an extra density case to test the performance under limiting conditions, used a well-known graph generation model and an often used layout algorithm, and evaluated a single, basic task (connectivity testing), which is an important, elemental and generally recurring task when inspecting graphs. We readily acknowledge, however, that other options and choices can and should be considered, which we further discuss in Section 8.

To evaluate the performance of the representations in terms of reading time and correctness, we ran an experiment in which participants performed path-readability (connectivity) trials, i.e., they had to answer whether or not there was a directed connection between a pair of nodes. We provide initial hypotheses regarding the performance with respect to varying graph densities and link lengths, and subsequently compare these with the outcome of a statistical analysis of reading times and correctness. Based on the analysis results, we provide practical recommendations on the use of directed-edge representations in the context of node-link graphs. This work significantly extends the previous work and gives a clearer choice of design alternatives for directed graphs.

3 Related Work

The following presents an overview of directed-edge representations that have been previously proposed, the majority being uniform, static representations as described in Section 3.1 used in the context of node-link graphs to depict directed connectivity. The use of edge representations that rely on – generally texture-based – animation is treated in Section 3.2. Section 3.3 then describes studies that have been performed to quantify how directed-edge representations perform in the context of node-link graphs.

3.1 Static Directed-Edge Representations

Half-lines are used by Becker *et al.* [4] to depict directed edges; a half-line from Node *A* to *B* is a straight-line connection in which only the first half of the line is drawn. Although half-lines reduce the amount of display space used to show links, they make it hard to determine where links end. Wong *et al.* [27] present their *GreenArrow* technique to balance the appearance of both a graph and its labels. The text label pertaining to a link (or one of its nodes) is drawn such that the text itself forms a tapered link between nodes. This removes the need to explicitly visualize the edge using a line-based representation. A color-coded representation is used by Holten [13] to indicate edge direction as running from Node *A* to Node *B* by gradually changing the color from green (*A*) to red (*B*) along the length of a link; different colors can be used as well.

Edge representations based on curvature (such as C; Figure 1f) have been used before in the context of *Arc Diagrams* by Wattenberg [24] and *ArcTrees* by Neumann *et al.* [19] Arcs, i.e., curved links, were used in *Arc Diagrams* to represent complex repetition patterns in string data and in *ArcTrees* to visualize hierarchical as well as non-hierarchical data relations. In case of such symmetric arcs, a curve's clockwise orientation is generally used to indicate direction. The aforementioned *GreenArrow* technique by Wong *et al.* [27] also uses additional curvature to indicate direction.

Instead of only using (counter-)clockwise orientation to indicate direction, Fekete *et al.* [9] add curvature bias to this. Their links are drawn as quadratic Bézier curves that vary the amount of curvature: high at the start and low at the end. This is one of the representations we evaluated in our second study (C_b; Figure 1k).

For completeness, other static representations were proposed in our initial study [15]: the single visual cue representations S, T, DL, LD, GR, and C (Figures 1a–1f) as well as the multiple visual cue representations TI, TC, IC, and TIC (Figures 1g–1j). Our follow-up study, furthermore, added C_b to this.

Apart from recommending the use of T overall because of its performance, we found that combining representations (at least the ones that were evaluated) did not result in significant performance gains [15]. Our follow-up study, furthermore, showed C_b providing no significant improvement over C [14]. Nielsen *et al.* [20] note that they were inspired by our initial-study results. For their *ABySS-Explorer* they use tapering to depict the orientation of "contigs," i.e., genome assemblies consisting of long contiguous sequences.

3.2 Animated Directed-Edge Representations

Most animated edge representations indicate edge direction based on the idea of having a dash pattern – generally implemented as a repeating "glyph" using texturing – move in the direction of the link along the length of a link. An example is the effective use of animated dash-pattern textures by Wegenkittl *et al.* [26] to show the flow motion of trajectories within analytical dynamical systems. This idea also served as the inspiration for the A and A_c representations (Figures 1l and 1m). Our follow-up study already showed that as far as performance is concerned, A is on par with T [14].

To address node and link clutter in dense node-link graphs, Ware *et al.* [22] evaluated motion-based highlighting techniques to provide effective access to such graphs. One of the evaluated techniques is an edge representation that uses animated sawtooth dash-pattern textures radiating out from the start node. For the path-readability and node-reachability tasks that participants performed, reading time and error rate were significantly lower when using animated-link highlighting than when using no highlighting. Ware *et al.* therefore argue that motion-based highlighting can be a valuable visual aid for understanding large graphs.

Bartram *et al.* [3] showed the potential of using animated causal overlays, e.g., animated links, on top of causality-depicting visualizations such as causal graphs. The idea was inspired by the fact that perceptual psychology showed that causality perception is a low-level visual event derived from certain types of motion.

Blaas *et al.* [5] present a spline-based way to smoothly visualize higher-order state transitions for the exploration of state sequences in large time-series. Because they noted that arrowheads distort the perception of a continuous spline, they opted for a texture-based approach using animated dash patterns to visualize edge direction. They are furthermore planning to continue their investigation into the visualization of directed edges using animated textures.

Animated links have also been added to the *Tulip* graph visualization library [1]. The dash patterns used by *Tulip* move along a straight line from start to end and consist of repeating "greater than" symbols ("$\rangle\rangle\rangle$") of alternating colors. Our glyph representations G and G_c (Figures 1n and 1o) are inspired by this technique.

3.3 Evaluation

Apart from our initial [15] and follow-up [14] studies, only few user studies have been performed that quantify how directed-edge representations perform in the context of node-link graphs. One of these is the user study performed by Wong *et al.* [27] for their *GreenArrow* approach. However, their focus was on how to balance the appearance of a graph and its labels, not on determining how well their approach works as a technique for depicting directed edges in comparison with other directed-edge representations.

4 Graphs and Directed-Edge Representations

This section provides details on our study setup. We include information on the choices that were made with regard to the generative graph model, the graph layout, the directed-edge representations, and the way in which node pairs were chosen during trials to ensure controlled testing of different link lengths.

4.1 Graph Model

To ensure the availability of enough different graphs with similar statistical properties for all of the participants and for all of the combinations of graph density, edge representation, and link length, we chose to generate random graphs using a graph generation model.

The graph model proposed by Ware *et al.* [23] used in our initial study [15] generates graphs as follows: "*For each node, form a directed edge to one or two other nodes, randomly selected, so that a single connection occurs p% of the time and two connections occur* $(100-p)\%$ *of the time*," where p controls the density. This model is part of a more focused class of real-world graph models and, as stated in [18], it would be more correct to generate real-world graphs using such a model instead of the random graph model proposed by Erdős *et al.* [8], who define a random graph as $G(n,p)$ in which each of the $\binom{n}{2}$ possible edges occurs with probability p. However, the node-degree distribution of the Ware-*et-al.*-graphs is not scale-free: the fraction $P(k)$ of nodes having k connections to other nodes is not modeled by $P(k) \sim k^{-\gamma}$ (typically $2 < \gamma < 3$). Many real-world graphs, however, *are* scale-free and exhibit small-world properties, including social, citation, and protein networks.

For our follow-up study [14], we therefore chose a graph generation model that produces graphs with small-world properties, e.g., the Barabási-Albert (BA) [2] or the Watts-Strogatz [25] model. We used the Network Workbench [21] to generate a collection of graphs with varying densities according to the BA model. After initial pilot runs that colloquially assessed the difficulty experienced by users when deciding if there is a connection between a pair of nodes, we settled for sparse, medium-density, and dense graphs (regarding on-screen density with respect to "ink" usage), with 60, 90 and 135 nodes, respectively ($60\times1.5=90, 90\times1.5=135$).

For our current study we reverted to the Ware-*et-al.*-model as in [15]. Although the latter does not provide scale-free graphs, it allows us to avoid the typical scale-free-model generation of high-degree nodes (subclusters); we contend that controlled testing with respect to density is more important for our purposes. Furthermore, we extended the density range to incorporate a limiting, "extra hard" case to investigate the performance under conditions which arise quite often for real-world graphs. The current study, thus, used sparse, medium-density, and dense graphs, with 70, 140 and 280 nodes, respectively ($70\times2=140, 140\times2=280$; see Figure 2).

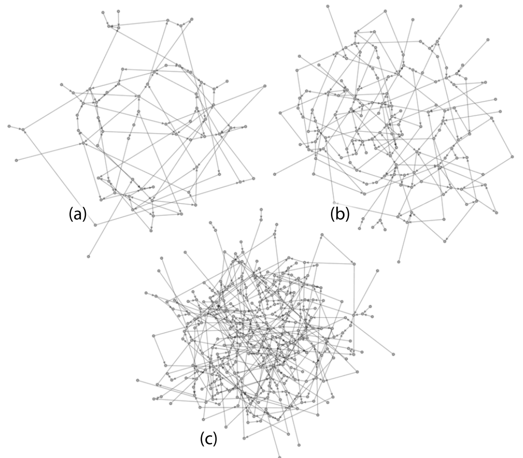

Figure 2: Graph densities used in the current study: (a) sparse (70 nodes), (b) medium-density (140 nodes), and (c) dense (280 nodes).

4.2 Graph Layout

There are many node-link-graph layout models available, the majority of which make use of force-directed node placement algorithms, such as Eades' spring-embedder model [6], the Kamada-Kawai model [16], or the Fruchterman-Reingold model [10]. We chose the widely used Fruchterman-Reingold (FR) algorithm to lay out our graphs. All layouts were generated by an FR implementation as provided by the *GraphViz* graph visualization software [7].

To ensure a clear distinction between edge lengths, we opted for two edge length classes – short and long – in the current study. For each generated graph, an edge was randomly assigned to be either short or long. The *GraphViz* FR algorithm allowed us to specify desired edge lengths during layout; we settled for long edges that were defined to be four times as long as short edges, ensuring clearly noticeable differences between short and long edges.

4.3 Directed-Edge Representations

Our current study evaluated T and A, the most promising candidates according to our previous studies, and a new texture-based "glyph" representation G that was added to further cover the design space. G – inspired by "sharp bend" traffic signs and the edges used by the *Tulip* visualization library [1] – uses repeating "greater than" symbols ("⟩⟩⟩") of two alternating shades of gray (Figure 1n).

We used a 20.1 inch Dell LCD display with a 16:10 width:height ratio and a resolution of 1680×1050 pixels to represent all of our graphs at 4× anti-aliasing on a white background. Given the size and resolution, our display provided approximately 99 DPI.

T was similar to the representation used in the previous studies: 0.05 inches and 0.005 inches wide at the start and end, respectively, and drawn in black at an opacity of 35% (Figure 1b).

A was also similar to the representation used in our previous studies: a moving dash pattern with a line width of 0.015 inches, a cycle length of 0.35 inches, 90% of which was occupied by a line (black, 50% opacity) and 10% of which was occupied by a gap, and moving at a speed of 0.17 inches per second (Figure 1l). Each link used a random phase to prevent distracting "bursts" of activity caused by multiple incoming/outgoing links around a node.

G used a static glyph pattern with a line width of 0.055 inches, a cycle length of 0.3 inches, 60% of which was occupied by a "greater than" symbol (">") (black, 50% opacity) and 40% of which was occupied by a gap. The "greater than" symbol of a non-stretched glyph mark furthermore used an acute angle of 75°.

T allows for edge length estimation by only having to look at part of an edge due to the varying width along its length. A similar advantage can be added to A and G by using dash/glyph pattern compression/stretching. Let $\bar{L}$ be the average edge length (calculated over all graphs). Let L be the length of an edge E. The amount C with which a pattern is compressed/stretched now becomes $C = (1-k) + k \cdot L/\bar{L}$, with $k \in [0,1]$. k determines the effect strength and was set to 0.5 to obtain a clearly discernible amount of compression/stretching while retaining as much edge uniformity as possible. Edges with $L > \bar{L}$ or $L < \bar{L}$ have their pattern stretched or compressed, respectively. Figures 1m (A_c) and 1o (A_c) show how this helps to estimate edge length.

G is also able to distribute visual clutter more evenly along an edge instead of concentrating it around a node (as is the case with arrowheads), thus providing users with multiple positions on an edge to assess its directionality.

4.4 Link-Length Selection

For each path-readability trial, a pair of nodes A and B was selected between which a directed link L – either short or long (as specified during layout as described in Section 4.2) – might be present. Since interesting difficulties caused by clutter and overlap occur more often in the central graph region, we ensured the selection of node pairs from this region. This was done by calculating a bounding circle for each layout and fitting a normalized, cut-off 2D Gaussian onto this with a value of 1 at the center and 0 (at 3σ) at the circular graph boundary. This value, denoted as $P_{center}(A,B) \in [0,1]$, represents the probability of a randomly chosen node pair consisting of nodes A and B being accepted based on the pair-midpoint position within the bounding circle. The probability of acceptance $P_{center}(A,B)$ depended on $d = \frac{A_{pos}+B_{pos}}{2} - C$, where C is the center of the bounding circle and A_{pos} and B_{pos} are the node positions. Similar average task difficulty across participants was realized by using a stable and predictable layout algorithm, by checking for degenerate layout cases prior to testing, and due to the substantial number of trials per participant.

5 Experiment – Comparing Directed-Edge Representations

5.1 Hypotheses

Based on our experience, we hypothesized that edge compression would improve the performance with respect to speed and correctness of both A and G and that, thus, animation in the form of A_c might now be able to surpass T. We also hypothesized that interaction effects would be present for the different graph characteristics. This led to the following hypotheses:

H1: Compression will improve performance for A and G since it provides a way to measure edge length "locally" by looking at only a part of an edge. This would allow participants to more easily exclude certain edges and help in cases where unconnected nodes would lie in the path of an edge;

H2: $A_c > T > A$. In our follow-up experiment, A already showed to be a competitive alternative for T for some graph types. On par with H1, we hypothesize A_c to outperform T; since A already rivaled T in terms of performance, we suspect the extra compression to be able to give A_c an advantage over T;

H3: All edge types are impacted in time and error performance for increasing graph density and edge length. We further hypothesize that G and T would be more affected in higher densities due to their increased use of "ink".

5.2 Design

We used a repeated-measures design with the following within-subjects independent variables: *edge representation* (T, A, A_c, G, G_c), *graph density* (sparse, medium density, dense), and *link length* (short, long). Each participant performed 10 blocks of 6 trials per edge representation. The order of edge representations was randomized based on a Latin Square and the presentation of *length* × *density* combinations was further randomized per block. Before each new representation, participants were allowed to rest and could continue to the next representation whenever they were ready. Experimental sessions lasted about 30 minutes including training. In summary, the design included:

3	graph densities	×
2	link lengths	×
10	repetitions	=
60	trials	×
5	edge representation	=
300	trials per participant	×
25	participants	=
7,500	**trials in total**	

5.3 Participants and Procedure

Twenty-five participants (18 male, 7 female) were recruited from two research institutions. They ranged in age from 22 to 49 years (median 30 years). All participants had normal or corrected-to-normal vision. 17 participants reported looking at node-link graphs at least weekly, the remaining participants reported monthly or

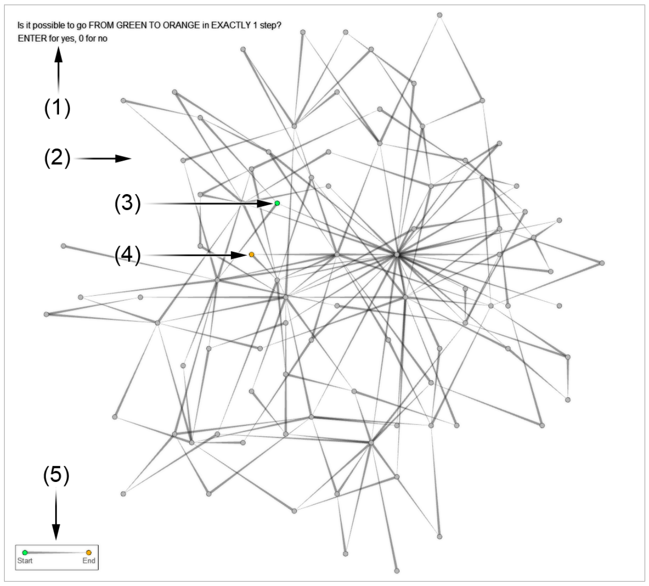
Is it possible to go FROM GREEN TO ORANGE in EXACTLY 1 step?
ENTER for yes, 0 for no
(1)
(2)
(3)
(4)
(5)
Start
End

Figure 3: Example view of a trial as presented to participants. Task instructions resided in the top-left corner (1), the graph in the screen center (2), and a legend on how to read the directed-edge representation in the lower-left corner (5). Test nodes were highlighted in green for the start node (3) and orange for the end node (4).

yearly exposure. 13 participants were students and 12 were non-students with varying technical as well as non-technical occupations. Participants were not paid for their involvement in the study.

During the experimental session, participants sat in front of a 20.1 inch Dell LCD display at a distance of approximately 50cm. Participants were first given an introduction to node-link graphs and the different edge representations. The experimenter then described the task, how to step through the trials, and how to record an answer, i.e., by pressing the "0" (no directed connection present) and "Enter" (directed connection present) keys on the keyboard; they pressed the "Space" key to start the trials for a new edge representation. Participants first conducted 30 practice trials to ensure that they were familiar with the experimental setup and procedure. If participants did not have any further questions, they continued with the experiment and filled out a post-session questionnaire to elicit qualitative feedback and demographic information.

5.4 Tasks

We used a path-readability task in which participants had to answer the following question throughout the study: *"Is it possible to go from the green node to the orange node in exactly one step?"* This task was chosen as it tests local readability of node connections and gives an indication of how well the chosen edge representation allows participants to infer directionality. Participants were instructed to be both as accurate and fast as possible. The correctness of their answer was shown to participants only during practice trials (immediately after a trial) and not during the actual experiment.

Each trial was shown to participants full-screen in three stages. During the first stage, participants saw an empty white screen for 400ms. During the second stage, we showed two randomly selected nodes A (green) and B (orange) to the participants for 600ms. This stage was introduced to ensure that participants did not spend time finding the two nodes in the graph. Finally, in stage three the complete graph (with A and B in the same positions) was displayed and timing for the trial was started.

A legend for the currently used edge representation was displayed in the lower-left corner of the screen. Participants were presented with a new graph for each trial and Nodes A and B were chosen such that there was a 50% chance of a connection being present; in 50% of the cases in which a connection was present, the connection also had the correct direction, i.e., there was a 25% overall chance of a connection being present and having the correct direction (from A to B). The chance of a bidirectional connection being present was not explicitly controlled for. The randomly selected graphs were generated and laid out as discussed in Sections 4.1 and 4.2 and link lengths were selected as discussed in Section 4.4. Figure 3 shows a typical test screen that participants saw in our study.

6 Results

This section includes the results of a statistical analysis of our study data. The completion times collected during the experiment were first log-transformed to comply with the normality assumption of the data analysis. Then, time was analyzed using a repeated-measures ANOVA. Timing data was recorded and analyzed at millisecond scale; average task times are reported here at second scale. The error data was analyzed using the non-parametric Friedman analysis of variance by ranks and Wilcoxon signed rank tests as the error data did not conform to the normality assumption. For each edge representation we removed the first block of trials from the analysis to account for learning effects.

In the following the five edge representations will be abbreviated as previously introduced: tapered (T), animated (A), animated compressed (A_c), glyph (G), and glyph compressed (G_c).

6.1 Overall Effect of Edge Representation

We observed task completion times ranging from $1.32s$ to $1.57s$. T was the fastest edge representation overall, followed by A_c and A. The picture was slightly different when analyzing overall correctness. Participants were most correct using both animated representations followed by T with correctness ranging from $93.2\% - 97.4\%$ overall. Participants performed both most slowly as well as least correctly with the glyph representations. Figure 4 gives an overview of both average time and error for each edge representation.

To understand significant relationships between the different representations according to time and error, we conducted a statistical analysis. We found a significant effect of edge representation on overall task completion time ($F(4,96) = 7.424; p = .002$). Post-hoc pairwise comparisons only showed a significant difference between T and G with $p < .001$. We also observed a significant effect of edge representation on correctness ($\chi^2(4) = 31.45; p < .001$). Wilcoxon signed rank tests showed several statistically significant results for pairwise comparisons. These are summarized in Table 1.

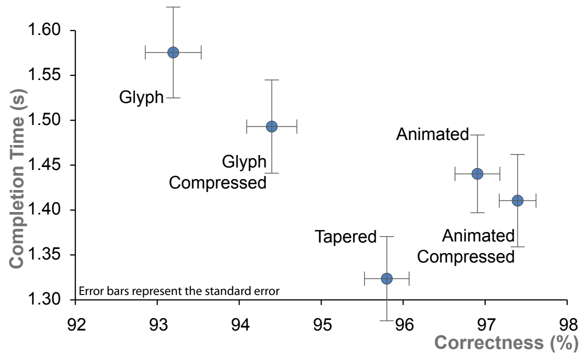

Figure 4: Average completion time (y-axis) and average correctness (x-axis) per representation. The best and worst techniques would be situated in the bottom-right and top-left corners, respectively.

	A_c-G_c	A_c-G	A_c-T	A-G_c	A-G	T-G
Z	−3.157	−3.834	−2.471	−3.215	−3.194	−2.96
p	.002	< .001	.013	.001	.001	.003

Table 1: Statistically significant results of pairwise comparisons for edge representations based on error (better technique on the left).

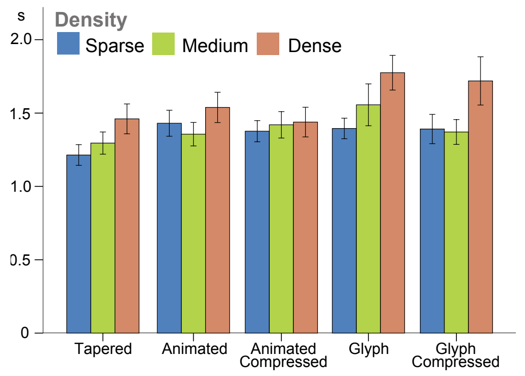

Figure 5: Average trial time per representation by graph density.

In summary, we observed that T performed well in terms of speed. However, we also showed that A_c is a competitive alternative to T as participants performed significantly more correctly than T using this technique. Also, average task completion times varied only within the range of .1s between both techniques. Glyph techniques, overall, did not compete well. We, thus, confirm our first hypothesis that edge compression improved the performance of A overall. We partially confirmed *H2* by showing that $A_c > T$ for overall task correctness. No significant difference was found between T and A for both speed and correctness.

6.2 Effects of Graph Density

To understand whether the overall effects would vary for graphs of different densities we investigated in more detail how each representation fared across the three densities we tested. Overall, tests showed a significant effect of graph density on task completion time $F(1.484, 35.627) = 16.112, p < .01$ (using Greenhouse-Geisser correction, $\varepsilon = .741$). Average completion times were 1.36s ($SD = .88$) for sparse graphs, 1.39s ($SD = 1.06$) for medium-density graphs, and 1.58s ($SD = 1.3$) for dense graphs. Post-hoc pairwise comparisons showed a significant effect between the dense graph and both others with $p < .001$. Error analysis also showed a significant effect between the three densities ($\chi^2(2) = 24.55; p < .001$). Post-hoc pairwise comparisons showed a significant difference for dense and the two other graphs with $p < .001$ each. Thus, we focused our analysis on both dense and sparse graphs.

Next, we looked at the effect of graph density with respect to the different edge representations. The analysis of task completion time showed a significant effect for $edge\ representation \times graph\ density$ ($F(8, 192) = 3.607, p = .001$). For sparse graphs T was the best technique and significantly outperformed G ($p < .006$), A ($p < .01$), and A_c (p.012). For dense graphs G was the worst technique and significantly slower than T ($p < .001$), A ($p < .024$), and A_c ($p < .004$). Interestingly, as can be seen in Figure 5, A_c was hardly affected by changes in density.

During the analysis of trial errors we observed a similar pattern. An overall significant difference for $edge\ representation \times graph\ density$ ($\chi^2(14) = 65.91; p < .001$) emerged. Further cross-comparisons per type of graph density showed the strongest significant effects for dense graphs. Here, G was significantly less correct than T ($Z = -3.097, p = .002$), A ($Z = -2.931, p = .003$) and A_c ($Z = -3.55, p < .001$). Similarly, G_c was significantly less correct than T ($Z = -2.311, p = .021$), A ($Z = -2.492, p = .013$), and A_c ($Z = -2.946, p = .003$). Interestingly, participants were significantly more correct with A_c compared to T ($Z = -2.097, p = .038$).

In summary, we saw that T performed fastest for sparse graphs but it was significantly less correct than A_c for dense graphs. A_c seemed to perform well across different graph densities and would be a good choice in programs were consistency of edge representation is important across a wide variety of edge representations. We, thus partially confirmed *H3* for density.

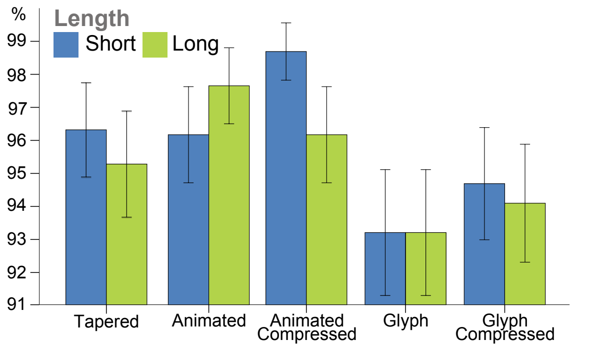

Figure 6: Average correctness per representation by edge length.

	Short edges					
	A_c-T	A_c-G	A_c-G_c	A_c-A	T-G	A-G
Z	-2.71	-3.97	-3.06	-2.35	-2.92	-2.53
p	.007	$< .001$	.002	.019	.003	$< .011$
	Long edges					
	A-G_c	A_c-G	A-T	A-G		
Z	-3.29	-2.21	-2.15	-2.97		
p	.001	.027	.032	.003		

Table 2: Significant differences for cross comparisons according to trial error for all five edge types.

6.3 Effects of Edge Length

We had further hypothesized that participants performance would be affected when judging connections of different lengths. Tests showed a significant effect of edge length on task completion time ($F(1, 24) = 37.501, p < .001$) with average task completion times of 1.34s ($SD = .9$) for short edges and 1.56s ($SD = 1.23$) for long edges. Post-hoc pairwise comparisons showed a significant effect between the two lengths with $p < .001$ in each case. No significant effect was observed between edge lengths in terms of correctness. Participants were 95.7% correct with short edges and 95.2% correct with long edges. Although the differences in error rate were not significant for the different edge lengths, the timing information shows that participants spent significantly longer for trials with longer edges indicating an increase in difficulty (see Figure 6).

The Friedman analysis of variance showed a significant effect of error rate for $edge\ representation \times edge\ length$ ($\chi^2(9) = 48.47; p < .001$). Table 2 shows the results of significant individual interactions. No significant effect of task completion time was observed for $edge\ representation \times edge\ length$.

In summary, A_c was the best representation for short edges as it lead to highest overall correctness. Both glyph representations did not fare well for both edge lengths. Interestingly, participants performed better with A for long edges than short edges and this representation was also significantly better than T for long edges.

6.4 Qualitative Feedback

The post-session questionnaire elicited feedback on subjective preferences and ratings for the five edge representations.

6.4.1 General Preference

The overall preference of participants showed clearly in the post-session questionnaire. Thirteen participants preferred T followed by A_c which was preferred by eight (Figure 7). 20 participants produced reasons for their choice of preferred technique. T was preferred because of its simple, clean, and clear layout (6 answers) and due to perceived speed and ease of edge recognition both for

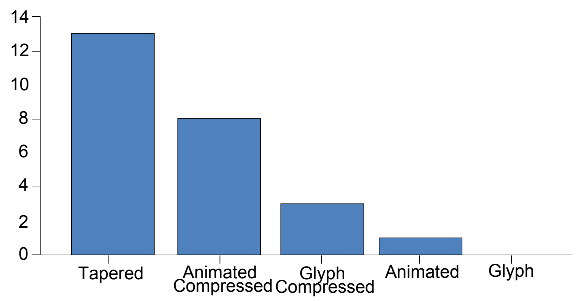

Figure 7: Number of participants preferring each representation.

direction and length of an edge (7 answers). The same general reasons were cited by the participants who preferred animated edges (7 answers overall). Participants also appreciated the reduced use of "ink" required for drawing animated edges (2 answers). Two participants who did not prefer A/A_c named visual interference as their main reason (2 answers). The glyph techniques were cited as being too noisy visually and thus difficult to read overall. Only three participants preferred G_c and nobody preferred G. The compressed techniques G_c and A_c were both more preferred than their counterparts G and A.

All techniques also scored favorably for the questions of whether participants considered them easy to read. Answers were recorded on a 7-point Likert scale ranging from 1: strongly disagree to 7: strongly agree. The answers ranged from a median of 5 (somewhat agree) for G, A, and A_c to 6 (agree) for both T and G_c.

In summary, participants showed a strong preference for T and A_c. Yet, they perceived all techniques to be generally easy to read. This is echoed in our observations that participants required very little learning time or explanation to understand the representations.

6.4.2 Self-rated Speed and Correctness

To capture possible differences in actual and perceived efficiency and effectiveness with a given edge representation, participants were asked to rank their perceived correctness and speed on a 7-point Likert scale. The scale again ranged from 1: strongly disagree to 7: strongly agree. All techniques scored highly. Participants agreed (median = 6) that T allowed them to be fast and somewhat agreed (median = 5) for the remaining representations. In terms of correctness, participants agreed (median = 6) that T, A, and A_c allowed them to be fast and also somewhat agreed (median = 5) for the two glyph techniques.

In summary, participants found all five edge representations to allow for fast and correct task completion.

6.4.3 Aesthetics

Participants were further asked to rate the representations on their ability to produce "nice-looking graphs" on a 7-point Likert scale. Participants rated T, A, and A_c highly (agree, median = 6) but gave a "somewhat disagree" rating for both glyph representations (G/G_c). Thus, the techniques which performed the best according to the statistical analysis were also rated highest in terms of aesthetics.

7 Discussion

7.1 Overall Recommendations

Overall, we partially confirmed our hypotheses: we saw that T and A_c performed best overall but there was a tradeoff in terms of speed vs. error. Participants performed most correctly with A_c but T showed to be faster on average. Both glyph techniques generally performed worst for all graph characteristics. We recommend to consider using either A_c or T and to take the design tradeoffs into account as discussed below. The only exception is the case of long edges where A was on average more correct than A_c and significantly more correct than T. Thus, in graphs of mostly long edges, A would also be a good alternative.

7.2 Effect of Pattern Compression

On average participants performed better in terms of correctness and task completion times for the compressed techniques compared to their non-compressed counterparts. Only in the case of long edges did A outperform A_c on average for correctness. However, for short edges A_c performed significantly better than A in terms of correctness. Participants also strongly preferred the compressed techniques, ranking them before the non-compressed ones. We, thus, conclude that pattern compression is a viable technique to use when edges are encoded with patterns (e.g., dashes or glyphs). They were well-liked by participants and also performed well. Compressed edge representations are not common in standard graph analysis packages but their use should be considered. From this study we conclude that being able to infer edge length from an edge representation is important and helpful for our test task.

7.3 Design Tradeoffs

All three main techniques incur tradeoffs in terms of their design and applicability. T is quite simple to implement and works for both printed media and screen display. Yet, T requires transparency and a varying thickness to be readable. If changing the link thickness or transparency according to an edge attribute is important, T cannot be used or could become less effective. Animated techniques, on the other hand, support varying thickness and variation of transparency, but require a screen setting and are harder to implement. Further research is also required to regarding different animation patterns. We proposed one that tested very well in comparison to T, but many different patterns are possible and may not support path readability in the same way (they may be better or worse). Pattern compression techniques are mainly valuable if there are clear edge length differences within a graph (which is often the case for small-world graphs), otherwise they provide only minor benefits.

The overall preferences for participants tended towards techniques that are simple and clean. Interestingly, compression techniques introduce additional visual variability among edges and we were initially worried that this may introduce additional clutter. Yet, we hypothesized that the drawbacks of this clutter would be outweighed by the ability to perceive edge length. Our initial worry about clutter was not confirmed for edge compression. Participants rated both A and G equally compared to their non-compressed techniques for aesthetics, performed better with the compressed techniques, and also preferred them.

8 Future Work

As mentioned in Section 2, the design space of directed links is large and we have only tried basic designs for reasons of feasibility. For animated edges it would, for instance, be interesting to measure the importance of speed (possibly based on edge length) now that we know that animation is helpful. Since compression provides an advantage as well, different compression schemes are also worth investigating. Another alternative worth considering is the placement of arrowheads in the middle instead of at the end of an edge to decrease overlap. As another example, since edge width can be freely varied for some representations, exploring how edge-width variation can be effectively combined with a directed-edge representation to encode an additional edge attribute is useful as well.

Now that we have the result for individual representations, we can also study how to use multiple representations within a graph for different edge types, how to use different representations in dense or sparse graph regions, or how to differentiate short and long links. Apart from that, representations can be combined (stacked) to form new ones, e.g., one that is tapered, glyphed, and animated at once. Although we found that stacking did not provide any direct benefits for combinations of tapering, intensity, and curvature [15], the results can very well be different for the currently tested representations, which makes the investigation worthwhile.

With respect to generalizability of the results, it would be of interest to explore other layouts as well. For example, stress majorization MDS [11] also yields good layouts and would therefore be worth investigating. Furthermore, apart from the single task that we have evaluated, there are several other tasks involved in graph visualization and the results of the tested representations might not generalize to all of them. Higher-level tasks, such as connectivity trials for paths spanning multiple nodes, or measuring the effect of the number of edges crossing a path should therefore be tested as well with respect to various directed-edge representations.

9 Conclusion

In this paper, we have reported on a controlled experiment comparing the readability of five directed-edge representations for node-link diagrams: tapered (T), animated (A), animated compressed (A_c), glyph (G), and glyph compressed (G_c). We tested these representation on graphs with three densities and edges of two different lengths. We tested one low-level connectivity task: showing two nodes, asking if the first node was connected to the second. We collected the completion time and the number of errors, as well as a questionnaire eliciting subjective feedback from participants.

The study showed that T and A_c were the best techniques overall and that both glyph representations are not to be recommended. T was the fastest technique and participants were significantly more correct using A_c. This study is the first that tested whether the ability to infer edge length from the edge representation would be important. Indeed, we conclude that both A_c and G_c techniques fared better on average than their non-compressed counterparts. Also, all three techniques which encoded edge length (including T) were the most preferred by our participants.

From this study and previous work, we can conclude that the best directed-edge representation among those studied for readability tasks are T and A_c. We provide recommendations in Section 7 for choosing a representation based on various tradeoffs.

Acknowledgements

We would like to thank all of the participants in our study for their time. This project is partly funded by the Netherlands Organization for Scientific Research (NWO) View programme under research grant no. 638.100.502 (*Expression of Interest Project*).

References

[1] D. Auber. Tulip: A Huge Graph Visualisation Framework. In P. Mutzel and M. Jünger, editors, *Graph Drawing Software*, pages 105–126. Springer-Verlag, 2003.

[2] A.-L. Barabási and R. Albert. Emergence of Scaling in Random Networks. *Science*, 286:509–512, 1999.

[3] L. Bartram and M. Yao. Animating Causal Overlays. *Computer Graphics Forum (Proceedings of EUROVIS)*, 27(3):751–758, 2008.

[4] R. A. Becker, S. G. Eick, and A. R. Wilks. Visualizing Network Data. *IEEE Transactions on Visualization and Computer Graphics (TVCG)*, 1:16–28, 1995.

[5] J. Blaas, C. Botha, E. Grundy, M. Jones, R. Laramee, and F. Post. Smooth Graphs for Visual Exploration of Higher-Order State Transitions. *IEEE Transactions on Visualization and Computer Graphics (TVCG; Proceedings of INFOVIS)*, 15(6):969–976, 2009.

[6] P. A. Eades. A Heuristic for Graph Drawing. In *Congressus Numerantium*, volume 42, pages 149–160, 1984.

[7] J. Ellson, E. R. Gansner, E. Koutsofios, S. C. North, and G. Woodhull. Graphviz and Dynagraph – Static and Dynamic Graph Drawing Tools. In M. Jünger and P. Mutzel, editors, *Graph Drawing Software*, pages 127–148. Springer-Verlag, 2003.

[8] P. Erdős and A. Rényi. On Random Graphs. *Publicationes Mathematicae*, 6:290–297, 1959.

[9] J.-D. Fekete, D. Wang, N. Dang, A. Aris, and C. Plaisant. Overlaying Graph Links on Treemaps. In *Poster Compendium of the IEEE Symposium on Information Visualization (INFOVIS)*, pages 82–83, 2003.

[10] T. M. J. Fruchterman and E. M. Reingold. Graph Drawing by Force-Directed Placement. *Software: Practice and Experience (SPE)*, 21(11):1129–1164, 1991.

[11] E. R. Gansner, Y. Koren, and S. North. Graph Drawing by Stress Majorization. In *Proceedings of the International Symposium on Graph Drawing (GD)*, pages 239–250, 2004.

[12] M. Ghoniem, J.-D. Fekete, and P. Castagliola. A Comparison of the Readability of Graphs Using Node-Link and Matrix-Based Representations. In *Proceedings of the IEEE Symposium on Information Visualization (INFOVIS)*, pages 17–24. IEEE, 2004.

[13] D. Holten. Hierarchical Edge Bundles: Visualization of Adjacency Relations in Hierarchical Data. *IEEE Transactions on Visualization and Computer Graphics (TVCG; Proceedings of INFOVIS)*, 12(5):741–748, 2006.

[14] D. Holten, P. Isenberg, J.-D. Fekete, and J. J. van Wijk. Performance Evaluation of Tapered, Curved, and Animated Directed-Edge Representations in Node-Link Graphs. Technical Report DH/PI/JDF/JvW/415, LaQuSo Laboratory for Quality Software, Eindhoven University of Technology, 2010.

[15] D. Holten and J. J. van Wijk. A User Study on Visualizing Directed Edges in Graphs. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems (CHI)*, pages 2299–2308. ACM, 2009.

[16] T. Kamada and S. Kawai. An Algorithm for Drawing General Undirected Graphs. *Information Processing Letters*, 31(1):7–15, 1989.

[17] R. Keller, C. M. Eckert, and P. J. Clarkson. Matrices or Node-Link Diagrams: Which Visual Representation is Better for Visualising Connectivity Models? *Information Visualization*, 5(1):62–76, 2006.

[18] G. Melançon. Just How Dense are Dense Graphs in the Real World? A Methodological Note. In *Proceedings of the AVI Workshop on BEyond time and errors: novel evaLuation methods for Information Visualization (BELIV)*, pages 75–81, 2006.

[19] P. Neumann, S. Schlechtweg, and M. S. T. Carpendale. ArcTrees: Visualizing Relations in Hierarchical Data. In *Proceedings of the Eurographics/IEEE VGTC Symposium on Visualization (EUROVIS)*, pages 53–60, 2005.

[20] C. B. Nielsen, S. D. Jackman, I. Birol, and S. J. M. Jones. ABySS-Explorer: Visualizing Genome Sequence Assemblies. *IEEE Transactions on Visualization and Computer Graphics (TVCG; Proceedings of INFOVIS)*, 15(6):881–888, 2009.

[21] NWB Team. Network Workbench Tool (Network Workbench: A Large-Scale Network Analysis, Modeling and Visualization Toolkit for Biomedical, Social Science and Physics Research). Indiana University, Northeastern University, and University of Michigan,2006.

[22] C. Ware and R. Bobrow. Motion to Support Rapid Interactive Queries on Node-Link Diagrams. *ACM Transactions on Applied Perception (TAP)*, 1(1):3–18, 2004.

[23] C. Ware and P. Mitchell. Visualizing Graphs in Three Dimensions. *ACM Transactions on Applied Perception (TAP)*, 5(1):1–15, 2008.

[24] M. Wattenberg. Arc Diagrams: Visualizing Structure in Strings. In *Proceedings of the IEEE Symposium on Information Visualization (INFOVIS)*, pages 110–116. IEEE, 2002.

[25] D. J. Watts and S. H. Strogatz. Collective Dynamics of Small-World Networks. *Nature*, 393(6684):440–442, 1998.

[26] R. Wegenkittl, M. E. Gröller, and W. Purgathofer. A Guided Tour to Wonderland: Visualizing the Slow-Fast Dynamics of an Analytical Dynamical System. Technical Report TR-186-2-96-11, Institute of Computer Graphics and Algorithms, Vienna University of Technology, 1996.

[27] P. C. Wong, P. Mackey, K. Perrine, J. Eagan, H. Foote, and J. Thomas. Dynamic Visualization of Graphs with Extended Labels. In *Proceedings of the IEEE Symposium on Information Visualization (INFOVIS)*, pages 73–80, 2005.