DoughNets: Visualising Networks Using Torus Wrapping

Kun-Ting Chen[1], **Tim Dwyer**[1], **Kim Marriott**[1], **Benjamin Bach**[2]
[1] Monash University, Australia, [2] University of Edinburgh, UK
{kun-ting.chen, tim.dwyer, kim.marriott}@monash.edu, bbach@ed.ac.uk

ABSTRACT

We investigate visualisations of networks on a 2-dimensional torus topology, like an opened-up and flattened doughnut. That is, the network is drawn on a rectangular area while "wrapping" specific links around the border. Previous work on torus drawings of networks has been mostly theoretical, limited to certain classes of networks, and not evaluated by human readability studies. We offer a simple interactive layout approach applicable to general graphs. We use this to find layouts affording better aesthetics in terms of conventional measures like more equal edge length and fewer crossings. In two controlled user studies we find that torus layout with either additional context or interactive panning provided significant performance improvement (in terms of error and time) over torus layout without either of these improvements, to the point that it is comparable to standard non-torus layout.

Author Keywords

Graph Visualization; Network Visualization; Torus Topology; User Study.

CCS Concepts

•**Human-centered computing** → **Graph drawings; Empirical studies in visualization;**

INTRODUCTION

Networks are widely used in the social sciences, life sciences, information technology and engineering to model complex relational data [20]. Visualisation is commonly used to understand and explore such networks. Node-link diagrams (commonly referred to as graphs) are the most common way to show these networks and a wide variety of automatic algorithms for finding readable and aesthetic layouts for these diagrams have been developed along with an extensive underlying mathematical theory [2].

Research has largely focused on layout of graphs on a 2D plane, for display on a printed page or standard screen. However, graph theoreticians have also considered embeddings of graphs on other topologies, such as the torus [15]. When a non-planar graph is drawn on the surface of a torus it is possible to avoid or reduce crossings between edges (links) by routing some of them through the hole of the torus instead of around the outside, as shown in Fig. 1. This figure also shows that it is possible to then "slice open" and flatten the surface of the torus so that it can be rendered on a screen or printed on paper. In such a flat rendering it is understood that edges which extend off the (for example) top, wrap around to the same horizontal position at the bottom of the drawing, and similarly for the left and right sides.

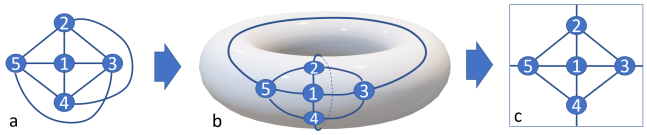


Figure 1. (a) A small complete graph of five nodes cannot be drawn without crossings on the plane, here we have a crossing between edges 2-4 and 3-5. (b) However, if drawn on the surface of a torus, we are able to route edge 3-5 around the outside of the torus while 2-4 goes through the hole such that they never intersect. (c) Slicing the torus open we are able to flatten it out and the topology is preserved, as long as the reader understands that edges that extend off the sides of the display wrap around to the same position on the other side.

Apart from crossing reduction, toroidal layouts may afford other benefits: e.g., greater angular resolution between edges connected to the same node and more uniform edge lengths. However, torus layout has so far been the preserve of theoreticians and has not been considered by the data visualisation community. We find it interesting that torus drawings of graphs, like those in Fig. 1, have not (to our knowledge), been seriously considered as a practical means of visualising networks. Is it because they are little known? Or is there a tacit belief that the topology is too difficult to understand?

There are some notable examples of video games using torus wrapping in a way that players intuitively understand the topology, e.g. Fig. 2. Is this intuitive understanding of such a topology also applicable to network visualisation? Three fundamental research issues must be addressed if torus layout is to become more widely used. We must:

RQ1: develop layout algorithms that take advantage of the extra flexibility of graph layout on the torus;

RQ2: determine how we can best visualise the layout of a node-link diagram on the surface of a torus on a piece of paper or 2D computer monitor; and

RQ3: determine what, if any, perceptual benefits graph layout on a torus has over standard layout on a 2D plane.

RQ2 is the most novel issue. While one could simply use linear projection to draw a 3D torus on the 2D plane, this causes significant occlusion. Instead, graph theoreticians typically show layouts on a torus by tearing the surface of the torus and projecting it onto the plane as illustrated in Fig. 1. The downside is that this can break edges which now wrap

CHI'20, April 25–30, 2020, Honolulu, HI, USA

DOI: https://doi.org/10.1145/3313831.3376180

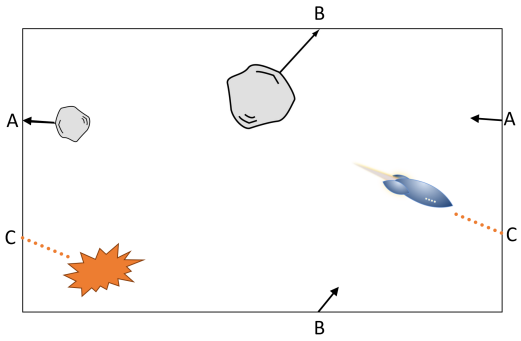


Figure 2. The video game "asteroids" is played on a flattened torus topology: here, we see that the path of the asteroid drifting to the left of the display will wrap around to the right at point A. Similarly, the asteroid drifting to the top will wrap around at B and the bullets of the space ship will wrap right-to-left at point C.

horizontally and/or vertically around the layout. To aid comprehension of such wrapping, in some illustrations of toroidal graph embeddings the layout is replicated on the edges of the original layout to provide partial or full context [12]; this means the edges are no longer broken but at the cost of replication. Another potential aid to understanding edge wrapping is interactive panning. We might hypothesise that the user seeing the detail wrap around as they pan will reinforce understanding, but again, to the best of our knowledge this hypothesis has never been tested.

Our contributions, therefore, are:

- We give a modified version of a stress minimisation algorithm that supports interactive human-guided layout of graphs on the torus (RQ1).
- A controlled, 24 participant study investigating whether providing partial or full context aids comprehension over a torus drawing without context. We found significant benefits to providing full-context over both partial and no-context for edge and path following tasks (RQ2).
- A second 24 participant study investigating whether interactive panning aided understanding of toroidal graphs. We found that panning improved performance of no-context torus drawings to the point where they were comparable to full-context torus drawings for path following tasks (RQ2).
- As part of our first study we compared the torus layouts with a standard force directed layout on a 2D plane (non-torus drawings). We found torus drawings were preferred over non-torus drawings for neighbour tasks and we found that with the addition of either full-context or panning we did not find significant differences between performance for torus and non-torus drawings for most edge, path following tasks. However, partial-context drawings and no-context, no-panning torus drawings were clearly worse than non-torus drawings, especially for path following tasks [RQ3].

The full set of study materials and evaluation results are available online: https://observablehq.com/@torus/doughnets-visualising-networks-using-torus-wrapping

BACKGROUND

Mathematicians have long been interested in the difficult combinatorial problems associated with identifying the smallest number of crossings required to embed a graph in the plane [18]. Topological graph theory extends problems such as these into embeddings of graphs onto *higher-genus* surfaces, essentially surfaces with holes through which some of the edges of a non-planar graph can be routed without intersecting other edges [15]. It would be fair to say that the majority of this work on higher-genus surface graph embedding has been theoretical or perhaps with application in areas such as circuit design. While some of this work has appeared in the Graph Drawing literature, which has cross-over between theory and network visualisation applications, graph drawing and visualisation researchers have not seriously considered practical network visualisation on higher-genus surfaces. Certainly, to the best of our knowledge, there is no available software that creates visualisations of arbitrary networks on higher-order surfaces. Algorithms capable of computing embeddings of certain types of graphs do exist [22, 4], however, they are restricted to graphs that permit an embedding without crossings on the torus. It seems they are rarely actually implemented, nor used in data visualisation.

The closest to network visualisation on higher-genus surfaces that we are aware of in the literature, is illustrations in the kind of theory papers described above. That is, illustrations of the properties of embeddings on surfaces. 3D drawings of tori or higher-genus surfaces are sometimes used, i.e. doughnuts, pretzels and other holey pastries. A "non-Euclidean spring embedder" that may be capable of producing 3D torus drawings has been proposed [11], but apparently not tested with a torus distance metric. Furthermore, it is not clear how this may be adapted to produce 2D wrapped-drawings. An interesting research question may be whether such 3D representations might be usefully used in immersive environments, but this question is beyond the scope of this paper. Rather, torus embeddings have the interesting property that they can also be represented in a two-dimensional diagram or visualisation, where it is understood that the left and right sides of the drawing connect, as do the top and bottom - as per the asteroids game mentioned earlier.

Our central interest, then, is whether these types of 2D toroidal drawings permit improved aesthetics and readability of the connectivity of the network, such that they can usefully be used in real-world network visualisation applications. While this particular question does not seem to have been asked by human-computer interaction and data visualisation researchers previously, there is a long tradition of evaluating the readability of different network visualisation techniques. We therefore have tools, like graph aesthetic metrics [17], that have been correlated with user preference and performance in readability tasks [21]. Number of edge-edge crossings, the first metric that we can demonstrate can be improved by toroidal layout, is certainly known to affect readability. Another metric that may benefit from a toroidal topology is the minimum angle between edges incident on nodes. However, we do not know if these improvements in metrics permitted by toroidal embeddings lead to an improvement in readability greater than any

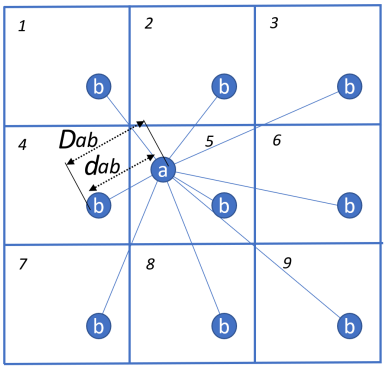


Figure 3. To compute the stress gradient information for node a with respect to another node b, we consider nine possible adjacencies. Here, the marked length D_{ab} corresponds to the ideal (graph theoretic related) distance between nodes a and b, while d_{ab} corresponds to the Euclidean distance in the current layout between the centres of nodes a and b. For this pair of nodes a, b, we will choose the wrapping that results in the smallest difference between ideal and actual Euclidean distance ($D_{ab} - d_{ab}$). Thus, we choose the adjacency between cells 4 and 5, resulting in a horizontal wrapping.

detrimental effect of, for example, the challenge of following edges that wrap around the boundaries of the drawing.

STRESS MINIMISING TORUS LAYOUT

Force-directed methods, which simulate a physical model of a graph with springy edges, are probably the most widely used layout algorithm in practical network visualisation applications. They can be applied to any type of graph, and generally do a reasonable job of untangling the crossings and normalising the lengths of edges in the graph where it is possible to do so. For very dense or scale-free graphs they are also well-known to produce "hair-ball" drawings that are difficult to read. There have been variants of force-directed algorithms proposed to produce drawings of graphs on non-Euclidean surfaces such as spheres, for example, by Kobourov and Wampler [11, 13]. However, while Kobourov and Wampler [11] mention the possibility of torus layout, it does not seem that it was actually tested. Furthermore, visualisation in this work is generally intended for 3D rendering of the surface on the screen or in virtual environments, not flattening to a 2D diagram as per Fig. 1(c).

Our first contribution is an adaptation of a fairly commonly used variant of the force-directed algorithm, defined as a minimisation of *stress* across the diagram. The objective *stress* function is defined for a graph with nodes V as:

$$stress = \sum_{(u,v)\in V\times V, u\neq v} \frac{1}{D_{uv}^2}(D_{uv} - d_{uv})^2 \qquad (1)$$

where D_{uv} is an ideal distance between a node pair (u,v), taken as the graph theoretic path length between them, and d_{uv} is the actual (Euclidean) distance between them in the drawing. Various methods for minimising *stress* are used. We follow [8] in using a gradient descent approach. Our approach is implemented as a modification of the *WebCoLa* [1] implementation of the algorithm from [8]. Note, that this technique also allows us to add constraints to avoid overlapping node labels.

We take a straight-forward approach to adapting the goal function (1) to the torus. At each iteration, we compute gradient contributions of each node, not just with respect to each other node, but with respect to the positions of each other node in the eight cells adjacent to the central cell in a 3×3 tiling corresponding to the possible adjacencies in a torus topology, as seen in Fig. 3. The positions of each node within each cell are computed by simple offset, i.e. in Figure 3 the position of node b in the top-left cell is simply b's x, y position in the central cell, minus the cell size, and so on for the other cells. This results in nine different sets of gradient contributions.

From these nine choices of gradient components for each node, we compute the one that results in the largest reduction of stress, and take this to be the entry in the gradient vector for that node. The resultant descent vector (computed from the gradient and a step size from the corresponding second derivative information, as per [8]) has horizontal and vertical components for each node with contributions from all other nodes e.g. for node a in Figure 3. If the descent vector takes a node beyond the cell boundaries, then the position for that node wrap around, as per asteroids rules.

Interactive Layout

As with all gradient descent approaches to optimisation of non-convex functions, it can happen that the layout can converge to a configuration corresponding to a poor local minimum of the stress function. In future, new approaches for initialising the layout to a better state may autonomously give a high-quality trade-off between aesthetic measures. For the layouts used in our study stimuli, however, we achieve this with an interactive, human-guided approach, made possible by our iterative stress minimisation approach. A user can drag nodes while layout is proceeding which can be useful to guide the layout away from poor local minima, but also to explore different configurations of torus layout and the different symmetries that they reveal, as seen in Fig. 4. This interactivity was useful in preparing stimuli for our controlled study (Section *Comparing Torus and Non-Torus Drawings*), which is not intended as an evaluation of the automatic layout algorithm, but rather an evaluation of the torus drawing paradigm in general as per RQ3, and for which we needed to control for the various aesthetic measures.

In summary, a human-guided stress minimising approach to generate a torus layout is as follows[1].

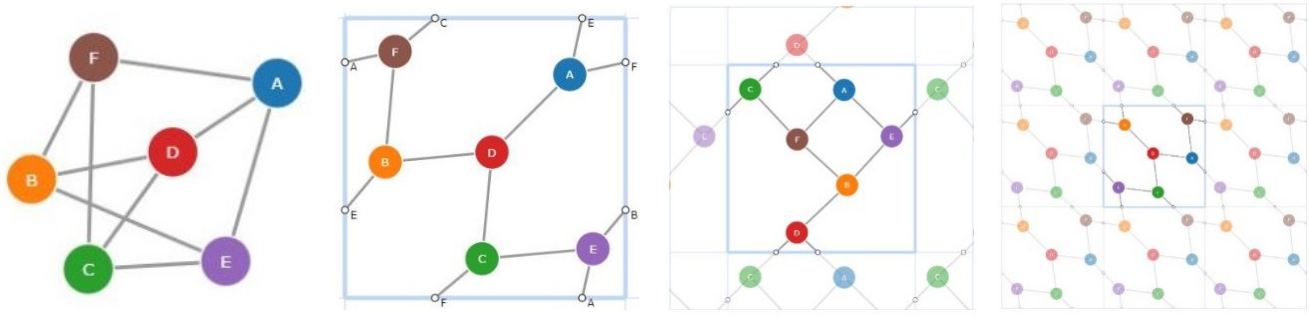
1. Automatically obtain an initial layout, starting with all nodes at the centre of the centre cell in a 3×3 grid.

2. Of the nine possible wrappings for each pair of nodes, choose the wrapping such that the torus distance is closest to the ideal (graph-theoretic related) distance, Fig.3.

3. For each pair of nodes, compute the partial derivative of the stress function for the chosen wrapping for that pair.

4. As per [8], we sum the partial derivatives across all pairs to give a gradient vector across all node positions.

[1]The detailed pseudocode to our method is available from https://github.com/Kun-Ting/WebCola



(a) NoTorus (b) NoContext (c) PartialContext (d) FullContext

Figure 4. The complete bi-partite k3,3 graph drawn using the four different techniques considered in this paper.

5. Also per [8] we use second derivative information to compute an optimal descent vector in the gradient direction.
6. After moving all nodes according to the descent vector some may have moved outside of the centre cell. These are translated back to the centre cell (e.g., by moving *b* in Figure 3 to its corresponding position in cell 5).
7. The above steps are repeated until convergence to a local minimum, i.e. movement falls below a predefined threshold.
8. User drags a node to a desired position.
9. The layout algorithm automatically adjusts the positions of all other nodes using aforementioned steps 2-7.

COMPARING TORUS AND NON-TORUS DRAWINGS

In this section, we compare different ways to render torus drawings of graphs with standard non-toroidal node-link diagram representations. We do this using both *i)* established metrics for assessing network layout aesthetic quality, and *ii)* some new metrics specific to torus drawings, as described below. We then prepared a corpus of 72 graphs using standard graph generation techniques that simulate real-world graphs as described in Section *Graph Corpus*. These were then laid out to balance the aesthetic measures using our interactive layout for both toroidal and non-toroidal conditions. Finally, we prepared three different styles of renderings of these graph layouts for use as stimuli in our two controlled studies.

Standard Layout Aesthetics Measures

The diagrams in our graph corpus were arranged to find a balance of a number of aesthetic measures. For a graph with node set V and edge set E we follow past work in considering the following graph layout aesthetics:

Edge Length Variance—Graphs with more uniform edge length have been found to be preferred by readers [6]. We compute edge length variance by first scaling the graph layout such that the average edge length is 1. Thereafter, we take the Edge Length Variance as the average squared deviation from 1 across all edges E.

$$\frac{1}{|E|}\sum_{e\in E}(1-l_e)^2 \tag{2}$$

where l_e is the length of edge e in the scaled drawing.

Minimum Angle—For a node v we take the ideal angle between adjacent edges incident to v to be $\theta_v = \frac{360^\circ}{deg(v)}$, where $deg(v)$ is the number of incident edges on v. Then, our Minimum Angle metric, is the average difference between this ideal angle of incidence and the actual minimum angle of incidence for edges incident to v in the drawing:

$$\frac{1}{|V|}\sum_{v\in V}\frac{|\theta_v - min\theta_v|}{\theta_v} \tag{3}$$

Edge-edge Crossings—The adverse effect of edge-edge crossings on readability of graphs is well studied [9]. Since we consider only straight-line drawings edge-edge crossings is a straightforward count of the number of times in each diagram that the line segments for pair of edges intersect. In all of the diagrams in our corpus we generally tried to keep crossing counts low by steering the stress minimisation algorithm, but truly crossing minimal drawings with straight-line edges are difficult; first, to identify, especially in larger graphs, and second—especially for the non-torus condition—it is difficult to nudge the stress layout into a state that leads to poor edge length variance. In general, we did not want to overly prioritise edge-edge crossing minimisation over the other metrics, as studies such as [10] indicate a balance of metrics is required.

Past work has found node-node overlaps and edge-node crossings significantly worsen graph readability [3]. As per [5], we include constraints in late iterations of the stress minimisation algorithm to prevent node overlaps. There are automatic processes for removing edge-node crossings [19, 8]. For the layouts in our diagram corpus, we remove node-edge crossings through a simple manual nudging, taking care not to overly disturb the other metrics.

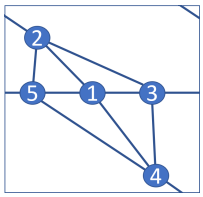
Torus Layout Aesthetics Measures

We define the following two measures for torus layouts, specifically.

Left-right and Top-bottom Wrapping—Straight-line edges lead the readers eye in an undemanding way from one node to its neighbours. However, where the edges wrap around the edges of a torus drawing, it seems reasonable to assume that finding the continuation of the edge on the other side of the diagram is a more demanding task. We report left-right and top-bottom wrapping separately, as we have controlled

the stimuli in our study such that their numbers are similar overall.

Corner Wrapping—When diagonal edges wrap near the corners of a torus diagram the edge must be split into three segments, as per the example to the right. In pilots for our first study we quickly realised that such corner-wrapping is especially confusing to readers. As described in Study 1, we therefore introduced additional training material to prepare participants for this case and controlled for corner-wrapping while generating our stimuli.



Graph Corpus

In this section we describe the test corpus of graphs, generated as described below, that we used in our studies. The layouts were generated using the human-guided stress minimising approach as described above and *WebCoLa* [1] layout for non-torus graphs, also with some manual adjustment. In general we tried to find layouts that balanced the crossings, minimum angle, and edge length variance statistics. However, there is a tradeoff. For example, in some cases fewer crossings may be possible (as in Fig. 4(a)), but it would come at a great cost to the other measures. As can be seen from the table, the torus embedding permitted significantly better layouts in terms of all three of these measures. However, it came at the new cost of wrapping. Table 1 provides summary statistics for these aesthetic measures.

We generated a sample corpus of 18 graphs rendered using the four different techniques for a total of 72 torus and non-torus drawings. The graphs were generated using algorithms designed to simulate real-world social networks and biological networks, using generators from NetworkX [16]. The majority of the graphs we used are Watts-Strogatz's small-world and Barabasi-Albert's scale-free graphs. We also used Erdos-Renyi's binomial network model to generate networks that balance the total number of wrappings. Therefore, we used one scale-free, small-world, and binomial model generator for creating graph instances for SMALL class. For MEDIUM class, we used two scale-free generators and one small-world generator, an example is shown in Fig. 5. For LARGE class, we used two scale-free generators and one small-world generator, an example is shown in Fig. 6.

Each class (SMALL, MEDIUM, LARGE) had a constant number of nodes (8, 11, and 15, respectively) and number of links in a tight range (12–18, 18–28, and 26–36).

TORUS RENDERING TECHNIQUES

In this section we comparatively evaluate four different visual representations for node-link diagrams. An example of each representation applied to the complete bi-partite graph known as *k3,3* is shown in Figure 4.

- NOTORUS is a conventional non-toroidal node-link diagram as laid out by a force-directed technique (WebCoLa [1]) with manual refinements to balance metrics, Fig. 4(a).
- NOCONTEXT is a torus drawing without any contextual tiling, as described below, Fig. 4(b). Note that the torus drawings used in our studies include labels at the boundaries indicating to which nodes wrapped edges are connected. The need for such labeling in torus drawings without additional context became clear from the pilots for our first study.
- PARTIALCONTEXT shows only partial context, that is, we see part of a repeated 3×3 tiling to show some torus adjacencies, Fig. 4(c).
- FULLCONTEXT shows the full 3×3 layout tiles (context) of a torus drawing, Fig. 4(d).

Note that the diagrams including context (PARTIALCONTEXT and FULLCONTEXT) are scaled to the same size in the figures in this paper, but in our studies they were shown to participants so that the central cell is the same size as the NOCONTEXT drawing; i.e., the area of FULLCONTEXT drawings shown to study participants were nine times that of NOCONTEXT drawings.

In Study 1 we evaluate static versions of these four representations. In Study 2 we introduce a simple panning interaction for the torus rendering techniques, giving us three more conditions: NOCONTEXT-PAN, PARTIALCONTEXT-PAN, and FULLCONTEXT-PAN.

STUDY 1: STATIC TORUS DRAWING READABILITY

In this first study, we investigate if our torus drawings are effective to solve low-level network analysis tasks and if they are more efficient than baseline drawings of node-link diagrams without torus edge wrapping. Thus the techniques in our study were exactly the same as described in the previous section. We were also interested in subjective user feedback and if the amount of visual *context* shown has an impact on effectiveness and efficiency of torus drawings.

Tasks

There are many tasks that users typically perform while analysing network visualisations [14]. For our study, we selected representative edge and path following tasks for network analysis [14] whose performance we believed would be influenced by torus edge wrapping.

- SHORTESTPATH: **What is the shortest path in terms of number of edges that need to be traversed between the nodes labelled *Start* and *End*?** Participants had to count the number of links between the marked nodes. We recorded participants' responses with multiple-choice questions with 8 options with answers of similar length but varying length.
- NEIGHBORS: **Identify all the friends (neighbouring nodes) of the orange node.** We recorded participants' response with multiple-choice questions with 8 options with similar ordered lists of nodes of various length.
- NODECOUNT: **Identify the total number of people (nodes) in the network**. To answer, we provided a slider with a range of 1 to 20. Error rate was obtained by calculating the absolute difference between participant's answers and actual answers divided by the actual answers.

Layout	*# Graphs*	*# Average crossings*	*# Average link length variance*	*# Average incidence angle*	*# Average Wrappings*	*# Average Corner wrappings*	*# Average Answers on wrappings*	*# Average Answers on corner wrappings*
Small-NoTorus	8	5.875	0.041	0.662	0	0	0	0
Small-Torus	24	0	0.023	0.376	2.729	0.25	5.625	0.25
Medium-NoTorus	6	13	0.050	0.650	0	0	0	0
Medium-Torus	18	0.778	0.035	0.429	3.917	0.833	3.5	0.5
Large-NoTorus	4	19.75	0.054	0.642	0	0	0	0
Large-Torus	12	4	0.049	0.479	4.75	1	3	0

Table 1. Average # crossings, link length variance, and incidence angle. Total number of torus wrappings and number of answers requiring a wrapping

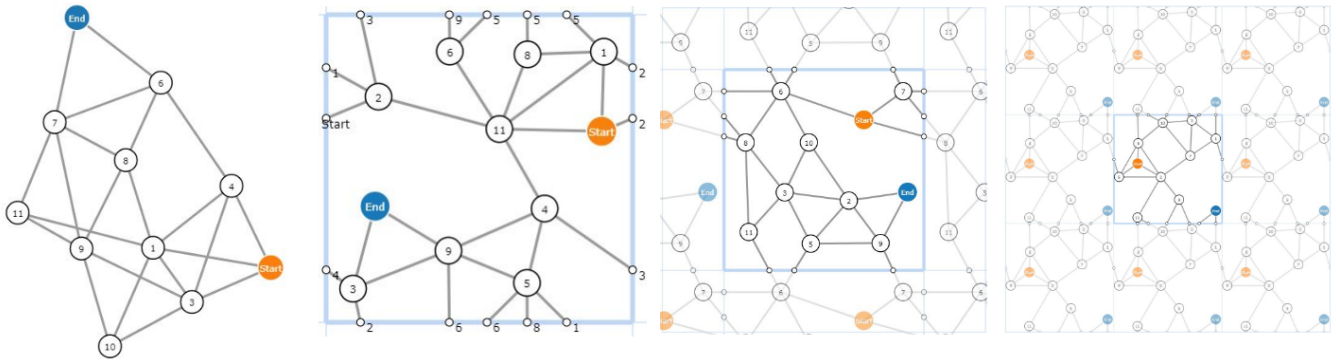


Figure 5. Example graph for our studies for MEDIUM. Techniques left-to-right are NOTORUS, NOCONTEXT, PARTIALCONTEXT and FULLCONTEXT.

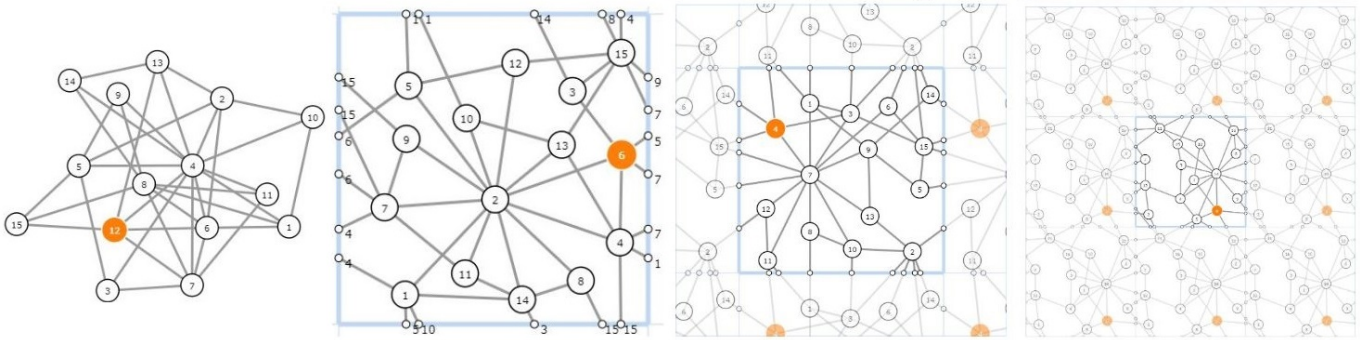


Figure 6. Example graph for our studies for LARGE. Techniques left-to-right are NOTORUS, NOCONTEXT, PARTIALCONTEXT and FULLCONTEXT.

- LINKCOUNT: **Identify the total number of relationships (links) of the network**. To answer, we provided a slider with a range of 1 to 20. Error rate was calculated the same way as for NODECOUNT.

Data sets

For all tasks, we chose graphs from one of the 18 graphs generated as described in Section *Graph Corpus*. Out of 18 trials in each condition in our study, there were 3 SMALL, 3 MEDIUM, 2 LARGE graphs (difficulty levels as defined above) used in SHORTESTPATH tasks, 3 SMALL, 3 MEDIUM, 2 LARGE in NEIGHBORS tasks, 1 SMALL graph in NODECOUNT, and 1 SMALL graph in LINKCOUNT. While the graph size (up to 15 nodes, 36 edges) is relatively small, as pointed out in [6, 7, 10], such small graphs already present a significant challenge to readability and present a suitable level of difficulty for path-following tasks in a study.

Hypotheses

Our predictions were pre-registered with the Open Science Foundation: https://osf.io/2e6bm.

Effect of layout

- **L1**: FULLCONTEXT has better task effectiveness (in terms of participant speed and error) than NOCONTEXT, and PARTIALCONTEXT across all task difficulties (RQ2).

- **L2**: FULLCONTEXT has better task effectiveness than NOTORUS (involves too many link crossings), NOCONTEXT

(requires too much mental wrapping), and PARTIALCONTEXT (requires certain mental wrapping) for difficult tasks (RQ2, 3).

- **L3**: PARTIALCONTEXT has better task effectiveness than NOCONTEXT across all task difficulties (RQ2).

- **L4**: NOTORUS has better (participant reported) task effectiveness than FULLCONTEXT for SMALL and MEDIUM graphs (RQ3).

Effect of tasks

- **T1**: Participants will perform better (in terms of participant speed and error) using FULLCONTEXT than NOCONTEXT and PARTIALCONTEXT on SHORTESTPATH and NEIGHBORS (RQ2).

Effect of size of graphs

- **D1**: Participants will perform better using FULLCONTEXT across all task difficulties than NOCONTEXT and PARTIALCONTEXT (RQ2).

Participant Preference

- **P1**: Participants will prefer FULLCONTEXT over either NOCONTEXT or PARTIALCONTEXT (RQ2).

- **P2**: Participants will prefer PARTIALCONTEXT over NOCONTEXT (RQ2).

- **P3**: Participants will report more confidence in using FULLCONTEXT than NOCONTEXT and PARTIALCONTEXT (RQ2).

- **P4**: Participants will report more confidence in using NOTORUS than FULLCONTEXT (RQ3).

Participants

We recruited 24 participants through email and snowball from our institute. 6 people were female while 18 were male. There were 2 participants aged below 20, 16 between 20-30, 5 between 30-40, and 1 greater than 50. While 5 participants never see social network diagram, the rest 19 responded they either seldom (15) or often (4) see network diagrams from course, textbook, or Internet.

Design and Procedure

We opted for a within-subject design study with 3 factors: 4 techniques $\times$ 3 difficulty levels $\times$ 4 tasks. We used a full-factorial design to balance for the 4 techniques (24 orderings). We used 18 trials with the number of graphs in each difficulty level (Table 1) + 10 training trials = 28 trials in each technique. For each technique, participants progressed through the same order of graph sizes from SMALL, MEDIUM to LARGE. Participants sat in front of a Dell 22-inch LCD screen. To collect visual focus of each representation, we equipped Tobii-pro X3-120 eye-tracking system to record visual focus of participants doing trials. We used a laptop with Intel I5 8350U (1.7GHz), Intel UHD Graphics 620 to run the website and control experiment.

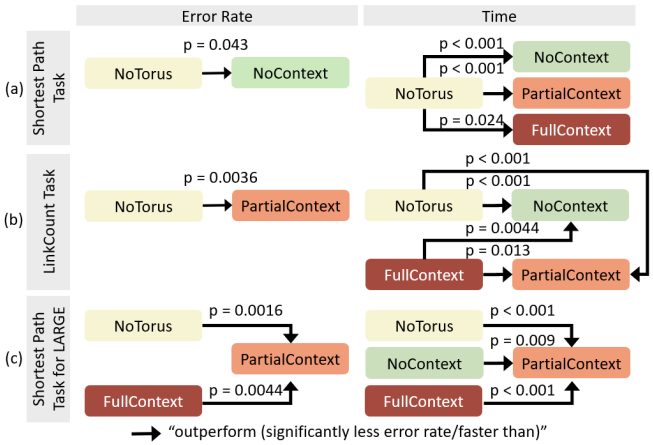


Figure 7. Study 1: Statistical results of performance comparisons between NOTORUS and Torus drawings under 95% confidence level.

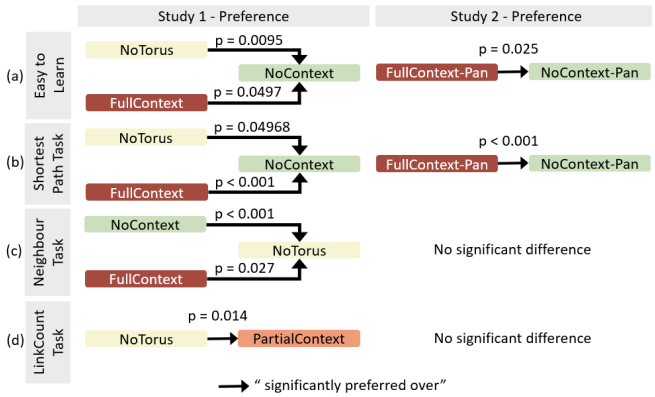


Figure 8. Study 1 and 2: Statistical results of preference differences (95% confidence level).

Results

All of the 24 participants completed 72 trials in the real (non-training) question set. They answered all the questions in tutorial and training sections correctly before entering the real question set. Therefore, we recorded performance of 1,728 trials. There were 29 significant differences in Study 1 as summarised below, based on 95% confidence levels. Since the distribution of the logarithm of completion time of each condition followed normal distribution, we used ANOVA repeated measures and Tukey's post-hoc pairwise comparison to test significance. Since the distribution of error rate and subjective rank of each condition did *not* follow normal distribution, we used Friedman's non-parametric test to test null-hypothesis and Nemenyi's post-hoc pairwise comparison to test significant difference. Significant differences are summarised in Fig. 7(error and time), and 8(participant preferences). Graphics with detailed results are found in Fig. 9.

For **SHORTESTPATH** and **LINKCOUNT**, our most significant results showed the following:

- NOTORUS significantly outperformed NOCONTEXT in error and time independent of size of graphs (as seen in Fig. 7(a), 7(b), boxplots in Fig. 9(a), 9(e), 9(g)) and outper-

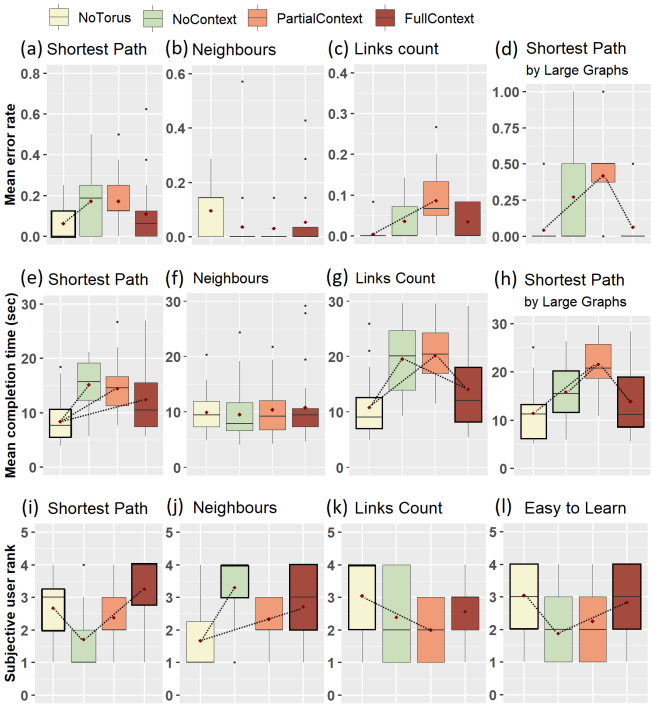


Figure 9. Study 1: Results for error, time, and subjective user rank by task. Higher rank indicates stronger preference. Dotted lines indicate significant differences for $p < 0.05$. The best significant results are highlighted in the border of bars.

formed PARTIALCONTEXT (as seen in Fig. 7(a), 7(b), with boxplots in Fig. 9(c), 9(e), 9(g);

- FULLCONTEXT significantly outperformed PARTIALCONTEXT in time independent of size of graphs (as seen in Fig. 7(b), with its boxplot in Fig. 9(g));
- For SHORTESTPATH by LARGE graphs, NOTORUS and FULLCONTEXT both significantly outperformed PARTIALCONTEXT in error and time (as seen in Fig. 7(c), with boxplots in Fig. 9(d), 9(h);
- Compared to NOCONTEXT, participants found it easier and reported greater confidence using both NOTORUS and FULLCONTEXT (as seen in Study 1 - Preference of Fig. 8(a), 8(b), with boxplots in Fig. 9(i), 9(l)). NOTORUS was significantly preferred over PARTIALCONTEXT (as seen in 8(d) with its boxplot in Fig. 9(k));

For **NEIGHBORS**, we found

- NOCONTEXT and FULLCONTEXT were significantly preferred over NOTORUS (Study 1 in Fig. 8(c), 9(j).
- This correlates with a weak trend for NOCONTEXT to outperform NOTORUS in error rate ($p = 0.23$) independent of size of graph, as seen in Fig. 9(b). The results of performance by graph size are omitted for space limitations and can be found in the study material web link.

Based on participants' qualitative feedback, they used NOCONTEXT with labelled display to quickly identify neighbours of a chosen node. NOCONTEXT drawing was (surprisingly to us) the most preferred torus display.

For **NODECOUNT**, results did *not* show any significant difference between techniques and most participants could correctly identify the number of nodes in all conditions, which suggests that most participants were not confused by the torus wrapping. The result is omitted and can be found in the study material web link.

Based on these results, we *rejected* the following hypotheses for RQ2: L3, T1 (for NEIGHBORS), P2, P4, and for RQ3: L2, L4 (for SMALL), P4. We accepted hypotheses for RQ2: L1, L2, T1 (for SHORTESTPATH), D1, P1, P3 and for RQ3: L4 (for MEDIUM).

Qualitative user feedback

The majority of participants mentioned that their preferences were dominated by the technique, independent from task and graph size. Participants favoured FULLCONTEXT for SHORTESTPATH over NOCONTEXT because it helped understanding edge wrapping and provided a great overview over the network. At the same time, they liked the cleanness of NOCONTEXT which has no repetition and extra information, which allowed them to concentrate the task.

Summary

In our first study, we generally found an improvement to torus drawing by adding full context to torus drawing. Our results can be summarised as follows:

1. For RQ3, FULLCONTEXT is as good as NOTORUS. Static NOCONTEXT or PARTIALCONTEXT torus is clearly worse than NOTORUS.
2. For RQ2, FULLCONTEXT is the best static torus layout. Static NOCONTEXT or PARTIALCONTEXT is the worst torus layout.
3. For NEIGHBORS, NOCONTEXT and FULLCONTEXT were both significantly preferred over NOTORUS in subjective user ranking (RQ3). Participants indicated that the NOCONTEXT technique appeared cleaner and would be better suited to show larger graph.

The role of context is crucial yet not entirely conclusive from our results. Result 3 above, regarding preference for NOCONTEXT over torus drawings with context, implies that the redundant information disturbs users, even as it assists them (as per Result 1 and 2). Moreover, context requires screen space. We therefore designed a 2nd study to investigate the effect of interactive panning across the three torus representations.

STUDY 2: TORUS DRAWINGS+PANNING

For our second study, we repeat the evaluation of the three torus conditions from Study 1 (NOCONTEXT, PARTIALCONTEXT, FULLCONTEXT): the visual representation in each of these techniques was exactly the same (e.g. Fig. 5 and 6). The sole interaction was panning, using the mouse. Since edge wrapping is not applicable to the node link representation we do not repeat trials for NOTORUS, but instead perform a

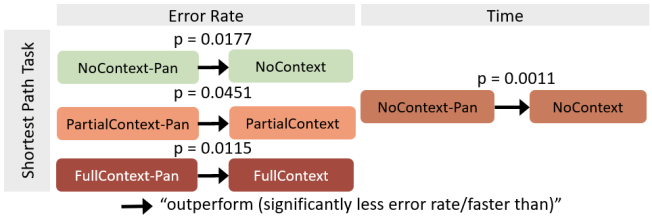


Figure 10. Study 2: Statistical results of performance comparisons between groups of static torus and interactive panning, under 95% confidence level.

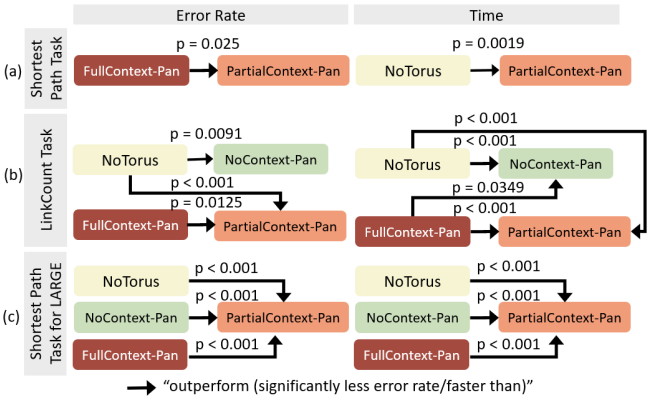


Figure 11. Study 2: Statistical results for performance of NoTorus and the torus+panning conditions, with 95% confidence level.

between-groups analysis of the Study 1 results for this condition, with those in Study 2. Tasks and graphs were also the same as in Study 1.

Hypotheses

Our predictions were pre-registered with the Open Science Foundation: https://osf.io/v3756.

- **I1**: NoContext-Pan has better task effectiveness than NoContext (RQ2).
- **I2**: PartialContext-Pan has better task effectiveness than PartialContext (RQ2).
- **P1**: Participants will prefer NoContext-Pan to FullContext-Pan (RQ2).

Participants, Design, and Procedure

We recruited a new set of 24 participants from local institute through email and snowball, none of which had participated in Study 1. 1 person were aged below 20, 17 between 20-30, 4 between 30-40, and 2 greater than 40. All of participants never or seldom see social network diagrams. Similar to Study 1, we used a within-subject design study with 3 factors: 3 techniques, 18 graphs, and 4 tasks. We performed a between-subject study comparing Interactive panning as in this study to the results of no-panning in Study 1. We use the same trials as previous user study and add interactive panning to the trials.

Results

All 24 participants successfully went through 54 trials in the real question set. They answered all the questions in tutorial

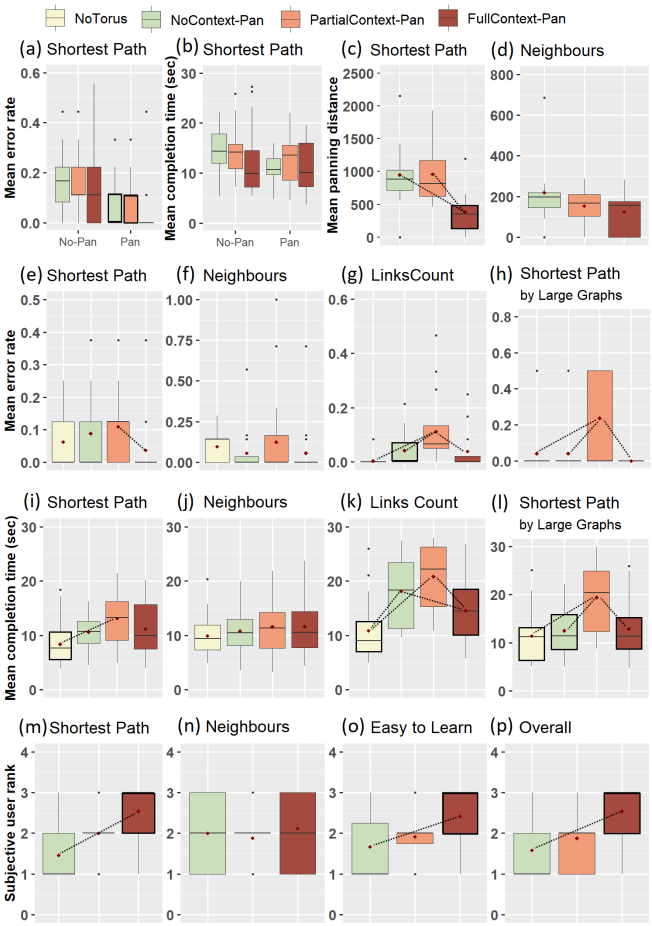


Figure 12. Study 2: Results for error, time, panning distance, and subjective user rank by task. Higher rank indicates stronger preference. Dotted lines indicate significant differences for $p < 0.05$. The best significant results are highlighted in the border of bars.

and training sections correctly before entering real question set. Therefore we recorded performance of 1,296 trials.

To test for significance, of panning on performance between panning and no-panning conditions, we used paired t-test. We used Wilcoxon's signed rank test to test significance difference in error rate between 2 groups. To test differences in performance between NoTorus in study 1 and torus with panning, we opted for ANOVA independent measures and Tukey's post-hoc pairwise comparison, as the logarithm of completion time followed normal distribution. We used Kruskal-Wallis's non-parametric test and Wilcoxon multiple pairwise comparisons to test error. To test panning distance between layout conditions we used ANOVA RM and Tukey's post-hoc pairwise comparison. Significant differences are shown in Fig. 10 (between groups of static and interactive panning), 11(for indiviual techniques) and 8 (subjective participant rankings). Graphics with detailed perfomance results are found in Fig. 12.

In contrast to Study 1, where there were significant differences between no-context and full-context torus or standard layout representations, with panning, we did not find any significant

differences between no-context and full-context or standard layout, independent of graph size for SHORTESTPATH task. This suggests that panning makes no-context torus techniques equally performant to full-context or standard layout, as participants pan more in the conditions where less context is given.

We found significant differences in the panning conditions for **SHORTESTPATH**:

- Independently of graph size, all torus panning significantly outperformed their non-panning counterpart in error (as shown in Fig. 10, 12(a)). NOCONTEXT-PAN significantly outperformed NOCONTEXT without panning in time (as shown in Fig. 10, 12(b));
- NOCONTEXT-PAN and PARTIALCONTEXT-PAN were panned significantly more than FULLCONTEXT-PAN, as seen in Fig. 12(c);
- With panning, NOCONTEXT-PAN, PARTIALCONTEXT-PAN and FULLCONTEXT-PAN all significantly outperformed PARTIALCONTEXT-PAN in error and time by LARGE (as seen in Fig. 11(c), 12(h), 12(l));
- NOTORUS significantly outperformed PARTIALCONTEXT-PAN in time (as shown in Fig. 11(a), 12(i)), and FULLCONTEXT-PAN significantly outperformed PARTIALCONTEXT-PAN in error (as shown in Fig. 11(a), 12(e)) independently of size of graphs.
- Compared to NOCONTEXT-PAN, participants found it easier, and reported greater confidence in using FULLCONTEXT-PAN (as seen in Study 2 - Preference of Fig. 8(a), 8(b), 12(m), 12(o)). Overall, FULLCONTEXT-PAN was significantly ($p = 0.0026$) preferred over NOCONTEXT-PAN, as seen in Fig. 12(p);

For **NEIGHBORS**, we did not find significant differences between any of the torus techniques, but for **LINKCOUNT**, both NOTORUS and FULLCONTEXT-PAN significantly outperformed NOCONTEXT-PAN and PARTIALCONTEXT-PAN in error and time (as shown in Fig. 11(b), 12(g), 12(k)), but FULLCONTEXT-PAN outperformed NOCONTEXT-PAN only with significant support in terms of time.

Based on these results, we rejected P1 and accepted I1 and partially accepted I2 (with significant support in terms of Error).

DISCUSSION

With respect to RQ3, our most significant results are 1) full-context or pannable full-context and no-context torus is as good as standard layout; 2) static or pannable partial-context torus is, despite being a drawing style most commonly used in the literature [11], the worst torus layout and clearly worse than standard layout. For RQ2, 1) full-context is the best static torus layout; 2) no-context + pan is as good as full-context.

Overall, we found that while torus topology allowed for significant improvements in the standard graph layout aesthetic measures (as per Table 1), the wrapping of edges at the sides of the diagram imposes a significant cost in terms of speed and accuracy in tasks requiring users to follow edges (as demonstrated in Study 1). However, this cost is mitigated by providing either more wrapped context in static diagrams (Study 1), or by allowing interactive panning (Study 2). Of the three torus representations shown in Study 2, the NOCONTEXT-PAN representation seems to have shown the greatest benefit from the introduction of panning, and indeed panning was used more in the NOCONTEXT-PAN condition than FULLCONTEXT-PAN. It is interesting that different levels of context (no, partial, full) has significant differences on torus readability. As a majority of users commented that they like the overview provided by full-context, it may be interesting to explore gaze distribution in different levels of context in future.

CONCLUSION AND FUTURE WORK

Our studies indicate that torus layout could be a practical technique (not worse than standard graph visualisation techniques) for the tasks tested, but that either redundant context or interactive panning are necessary. The graphs tested were automatically generated using algorithms designed to simulate naturally occurring graphs, but we would like to further evaluate the torus representations in a real-world application and see if it is usable by domain experts. We recognise that testing other types of tasks (such as cluster identification) on larger graphs is an important next step - but it is beyond the scope of this initial study which focuses on precise graph readability and path/edge following. We would also like to see if torus drawing works well for graphs larger than those in our studies (the largest had 15 nodes and 36 edges). We hope that the reduced clutter torus diagrams may work well for tasks particularly important to large networks, and those with complex structures that can benefit from a relaxation of the structure and less line crossings.

Technically speaking, our stress-minimising torus layout method is the first method we are aware of that is able to layout all graphs (not just graphs limited to a particular genus) on a torus topology. While we can force it to converge we cannot make strong guarantees that it converges to a local optimum. The interactive nature of the algorithm means that a user can guide it to a quite reasonable layout, certainly layouts that were good enough for our study. However, we are interested to see if the combinatorial techniques for layout of restricted classes of graphs on the torus, can be adapted to help us find good starting conditions for our stress-minimising torus layout in order to create a robust and completely autonomous high-quality torus layout.

REFERENCES

[1] 2015. WebCoLa: Constraint-Based Layout in the Browser. (2015). `https://ialab.it.monash.edu/webcola`

[2] Giuseppe Di Battista, Peter Eades, Roberto Tamassia, and Ioannis G. Tollis. 1998. *Graph Drawing: Algorithms for the Visualization of Graphs* (1st ed.). Prentice Hall PTR, Upper Saddle River, NJ, USA.

[3] Chris Bennett, Jody Ryall, Leo Spalteholz, and Amy Gooch. 2007. The aesthetics of graph visualization. *Computational Aesthetics* 2007 (2007), 57–64.

[4] Christian A Duncan, Michael T Goodrich, and Stephen G Kobourov. 2011. Planar drawings of higher-genus graphs. *Journal of Graph Algorithms and Applications* 15, 1 (2011), 7–32.

[5] Tim Dwyer, Yehuda Koren, and Kim Marriott. 2006. IPSep-CoLa: An incremental procedure for separation constraint layout of graphs. *IEEE Transactions on Visualization and Computer Graphics* 12, 5 (2006), 821–828.

[6] Tim Dwyer, Bongshin Lee, Danyel Fisher, Kori Inkpen Quinn, Petra Isenberg, George Robertson, and Chris North. 2009. A comparison of user-generated and automatic graph layouts. *IEEE Transactions on Visualization and Computer Graphics* 15, 6 (2009), 961–968.

[7] Tim Dwyer, Kim Marriott, Falk Schreiber, Peter Stuckey, Michael Woodward, and Michael Wybrow. 2008b. Exploration of networks using overview+ detail with constraint-based cooperative layout. *IEEE Transactions on Visualization and Computer Graphics* 14, 6 (2008), 1293–1300.

[8] Tim Dwyer, Kim Marriott, and Michael Wybrow. 2008a. Topology preserving constrained graph layout. In *International Symposium on Graph Drawing*. Springer, 230–241.

[9] Weidong Huang, Peter Eades, and Seok-Hee Hong. 2009. Measuring effectiveness of graph visualizations: A cognitive load perspective. *Information Visualization* 8, 3 (2009), 139–152.

[10] Steve Kieffer, Tim Dwyer, Kim Marriott, and Michael Wybrow. 2015. Hola: Human-like orthogonal network layout. *IEEE Transactions on Visualization and Computer Graphics* 22, 1 (2015), 349–358.

[11] Stephen G Kobourov and Kevin Wampler. 2005. Non-Euclidean spring embedders. *IEEE Transactions on Visualization and Computer Graphics* 11, 6 (2005), 757–767.

[12] William Kocay and Donald L Kreher. 2016. *Graphs, algorithms, and optimization*. Chapman and Hall/CRC.

[13] Oh-Hyun Kwon, Chris Muelder, Kyungwon Lee, and Kwan-Liu Ma. 2016. A study of layout, rendering, and interaction methods for immersive graph visualization. *IEEE Transactions on Visualization and Computer Graphics* 22, 7 (2016), 1802–1815.

[14] Bongshin Lee, Catherine Plaisant, Cynthia Sims Parr, Jean-Daniel Fekete, and Nathalie Henry. 2006. Task Taxonomy for Graph Visualization. In *Proceedings of the 2006 AVI Workshop on BEyond Time and Errors: Novel Evaluation Methods for Information Visualization (BELIV '06)*. ACM, New York, NY, USA, 1–5. DOI: `http://dx.doi.org/10.1145/1168149.1168168`

[15] Bojan Mohar and Carsten Thomassen. 2001. *Graphs on Surfaces*. Vol. 10. JHU Press.

[16] Python Package. 2019. NetworkX graph library. `https://networkx.github.io`. (2019). Accessed: 2019-09-20.

[17] Helen C Purchase. 2002. Metrics for graph drawing aesthetics. *Journal of Visual Languages & Computing* 13, 5 (2002), 501–516.

[18] Marcus Schaefer. 2013. The graph crossing number and its variants: A survey. *The Electronic Journal of Combinatorics* 1000 (2013), 21–22.

[19] Paolo Simonetto, Daniel Archambault, David Auber, and Romain Bourqui. 2011. ImPrEd: An Improved Force-Directed Algorithm that Prevents Nodes from Crossing Edges. In *Computer Graphics Forum*, Vol. 30. Wiley Online Library, 1071–1080.

[20] Colin Ware. 2012. *Information Visualization: Perception for Design*. Elsevier.

[21] Colin Ware, Helen Purchase, Linda Colpoys, and Matthew McGill. 2002. Cognitive measurements of graph aesthetics. *Information Visualization* 1, 2 (2002), 103–110.

[22] Jiahua Yu. 2014. *A Practical Torus Embedding Algorithm and Its Implementation*. Ph.D. Dissertation. Applied Sciences: School of Computing Science.