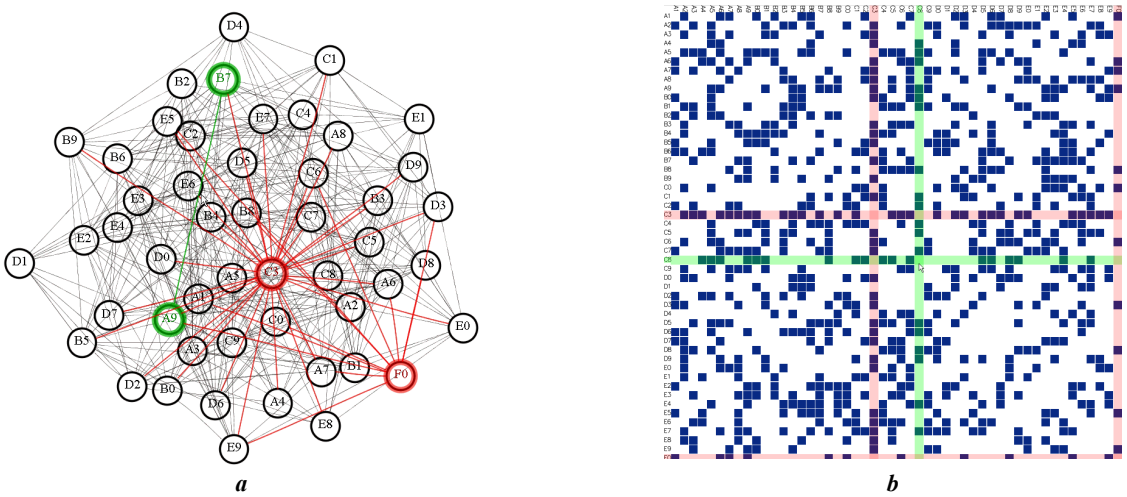
A Comparison of the Readability of Graphs Using Node-Link and Matrix-Based Representations

Mohammad Ghoniem[1]
Ecole des Mines de Nantes
4 rue Alfred Kastler. B.P.20722
44307 NANTES Cedex 3

Jean-Daniel Fekete[2]
INRIA Futurs/LRI
Bât 490, Université Paris-Sud
91405 Orsay Cedex

Philippe Castagliola[3]
Ecole des Mines de Nantes & IRCCyN
4 rue Alfred Kastler. B.P.20722
44307 NANTES Cedex 3



a *b*

Figure 1 Two visualizations of the same undirected graph containing 50 vertices and 400 edges. The node-link diagram a) is computed using the "neato" program and the matrix representation b) is computed using our VisAdj program.

ABSTRACT

In this paper, we describe a taxonomy of generic graph related tasks and an evaluation aiming at assessing the readability of two representations of graphs: matrix-based representations and node-link diagrams. This evaluation bears on seven generic tasks and leads to important recommendations with regard to the representation of graphs according to their size and density. For instance, we show that when graphs are bigger than twenty vertices, the matrix-based visualization performs better than node-link diagrams on most tasks. Only path finding is consistently in favor of node-link diagrams throughout the evaluation.

Additional keywords: Visualization of graphs, adjacency matrices, node-link representation, readability, evaluation.

Categories and Subject Descriptors: H.5 [Information Interfaces and Presentation]: User Interfaces – Evaluation; I3 [Computer Graphics]: Picture/Image Generation – Display Algorithms.

1 INTRODUCTION

Node-link diagrams have often been used to represent graphs. In the graph drawing community, many publications deal with layout techniques complying with aesthetic rules such as minimizing the number of edge-crossings, minimizing the ratio between the longest edge and the shortest edge, and revealing symmetries [2]. Most works strive to optimize algorithms complying with such rules, but they scarcely try and validate them from a cognitive point of view. Recently, Purchase et al. tackled this problem through on-paper [13] and online [10, 11] experiments. These works involved small handcrafted graphs (graphs with 20 vertices and 20 to 30 edges), five aesthetic criteria and eight graph layout algorithms. They point out that while some aesthetic criteria taken separately may improve the perception of the graph at hand, one cannot say that an algorithm brings about such an improvement. Moreover, Ware and Purchase set up a study aiming at the validation of some aesthetic properties put forth in the graph drawing community, such as the influence of good continuity on the perception of paths [13]. In the Information Visualization (Infovis) community, many node-link variants have been experimented, both in 2D and 3D [5, 8]. However, as soon as the size of the graph or the link density increases, all these techniques are facing occlusion problems due to links overlapping (Figure 1a). Thus, it becomes difficult for users to visually explore the graph or interact with its elements.

Conversely, matrix-based visualizations of graphs eliminate altogether occlusion problems and provide an outstanding potential, despite their lack of familiarity to most users. In this paper, we present an evaluation comparing the two representations in order to show their respective advantages with regard to a set of generic analysis tasks.

[1]Mohammad.Ghoniem@emn.fr
[2]Jean-Daniel.Fekete@inria.fr
[3]Philippe.Castagliola@emn.fr

IEEE Symposium on Information Visualization 2004
October 10-12, Austin, Texas, USA
0-7803-8779-1/04/$20.00 ©2004 IEEE

2 THE MATRIX-BASED VISUALIZATION OF GRAPHS

The matrix-based visualization of graphs relies from a formal standpoint on the fact that a graph may be represented by its connectivity matrix which is a matrix of Booleans whose rows and columns represent the vertices of the graph. When dealing with directed graphs, columns represent the origin of edges and the lines represent their endpoint vertices, although conventions may vary. When two vertices are connected, the cell at the intersection of the corresponding line and column contains the value "true". Otherwise, it contains the value "false". Boolean values may be replaced with valued attributes associated with the edges that can provide a more informative visualization (Figure 1b).

The matrix-based representation of graphs offers an interesting alternative to the traditional node-link diagrams. In [4], Bertin shows that it is possible to reveal the underlying structure of a network represented by a matrix through successive permutations of its lines and columns. In [3], the authors visualize the load distribution of a telecommunication network using a matrix but most of their effort aims at improving the display of a node-link representation such as displaying half-links or postponing the display of important links to minimize occlusion problems. More recently, in [7], the authors implemented a multi-scale matrix-based visualization representing the call graph between software components in a big medical imagery application. In [6], we have shown that a matrix-based representation can be used to effectively grasp the structure of a co-activity graph and assess the activity taking place across time whereas the equivalent node-link representation was unusable. This work was specifically applied to monitoring constraint-oriented programs.

3 COMPARISON OF REPRESENTATIONS

The comparison of two visualization techniques can only be carried out for a set of tasks and a set of graphs. The list of tasks that are useful or important with regard to graph exploration is endless. Indeed, one can realize this fact by choosing a concrete example, like a graph computed from a Web site and enumerating all the tasks that one can achieve or wish to achieve on such a graph. In order not to venture into this bottomless sink, we tackled the problem by considering the most generic tasks of information visualization and we adapted them to the visualization of graphs. We believe that the readability of a representation must be related to the ability of the user to answer some questions about the overview. As far as graphs are concerned, some questions may bear on their topology while other questions may concern attributes related to that topology. The most generic questions related to the topology of a graph – i.e. the ones independent of the semantics of data – bear on its vertices, links, paths and sub-graphs.

Basic characteristics of vertices: one may be interested in determining the number of vertices (their cardinality), outliers, a given vertex (by its label), and the most connected or least connected vertices.

Basic characteristics of paths: the number of links, the existence of a common neighbor, the existence of a path between two nodes, the shortest path, the number of neighbors of a given node, loops and critical paths.

Basic characteristics of subgraphs: one may be interested in a given subgraph, all the vertices reachable from one or several vertices (connected sets) or a group of vertices strongly connected (clusters).

Therefore, comparing the readability of graph representations should, in principle, take all these characteristics into account in order to determine the tasks that are more easily performed with a matrix-based representation and the ones for which it is more appropriate or more reasonable to use a node-link representation. This article presents a comparative evaluation of readability performed on a subset of these generic tasks due to time constraints.

3.1 Readability of a graph representation

One can reasonably define the readability of a graphic representation as the relative ease with which the user finds the information he is looking for. Put differently, the more readable a representation, the faster the user executes the task at hand and the less he makes mistakes. If the user answers quickly and correctly, the representation is very readable for the task. If the user needs a lot of time or if the answer he provides is wrong, then the representation is not well-suited for that task.

In our evaluation, we selected the following generic tasks:

- Task 1: approximate estimation of the number of nodes in the graph, referred to as "nodeCount".
- Task 2: approximate estimation of the number of links in the graph, referred to as "edgeCount".
- Task 3: finding the most connected node, referred to as "mostConnected".
- Task 4: finding a node given its label, referred to as "findNode".
- Task 5: finding a link between two specified nodes, referred to as "findLink".
- Task 6: finding a common neighbor between two specified nodes, referred to as "findNeighbor".
- Task 7: finding a path between two nodes, referred to as "findPath".

Readability also depends on the concrete graphs, their familiarity to users, their meaning and the layout used to visualize them. In our evaluation, we only compared random graphs which are meaningless and never familiar to user (e.g. equally unfamiliar), focusing only on abstract characteristics of graphs. We choose a popular graph layout program called "neato", part of the GraphViz[1] package to compute the node-link diagram. It could be argued that another layout program would provide a more readable layout according to our tasks. This is certainly true for actual figures but we believe that the trends would be similar when increasing the size and density of graphs.

3.2 Preliminary Hypotheses

The traditional node-link representation suffers from link overlapping – interfering with neighborhood finding and link counting – and link length – interfering with neighborhood finding. Moreover, some tasks involving sequential search of graph elements, such as node finding by name, are increasingly difficult when the number of nodes becomes large since, in general, nodes are not laid out in a predictive order. Hence, we expect the number of nodes and the link density to influence greatly the readability of this representation. We define the link density d in a graph as:

$$d = \sqrt{\frac{l}{n^2}}$$

where l is the number of links and n the number of nodes in the graph. This value varies between 0 for a graph without any edge to 1 for a fully connected graph. In graph theory, the density of a graph is usually taken as the ratio of the number of edges by the number of vertices but this definition – although topologically meaningful – is not scale invariant since the number of potential edges increases in the square of the number of vertices.

3.3 Predictions

The matrix-based representation has two main advantages: it exhibits no overlapping and is orderable. We therefore expect tasks involving node finding and link finding to be carried out more easily. Counting nodes should be equally difficult on both repre-

sentations, unless nodes become cluttered on node-link diagrams. Counting links should be easier on matrices since no occlusion interferes with the task. Finding the most connected node should perform better on matrices for dense graphs because, on node-link diagrams, links starting or ending at a node are hard to discriminate from links crossing the node.

On the other hand, when it comes to building a path between two nodes, node-link diagrams should perform better; matrix-based representations are more complex to apprehend because nodes are represented twice (once on both axes of the matrix), which forces the eye to move from the line representing a vertex to its associated column back and forth, unless a more appropriate interaction is provided. Lastly, we believe that node-link diagrams are suitable, and therefore preferable to the less intuitive matrix representation, when dealing with small sized graphs.

3.4 Experimental setup

3.4.1 The data

In order to test our hypotheses, we experimented with graphs of three different sizes (20 vertices, 50 vertices and 100 vertices) with three different link densities (0.2, 0.4 and 0.6) for each size, that is to say a total of nine different graphs (Table 1). In order to avoid any bias introduced by some peculiarity of the chosen data, we opted for random undirected graphs generated by the random graph server located at the ISPT [10]. Moreover, in order to eliminate any ambiguity with regard to task 3, which consists in finding the most connected node, we added an extra 10% of links to the most connected node in these graphs. When several nodes had initially the highest degree, one of them was chosen at random and received an additional 10% of links. The distribution of additional links was also done at random.

size\density	0.2	0.4	0.6
20	graph 1	graph 2	graph 3
50	graph 4	graph 5	graph 6
100	graph 7	graph 8	graph 9

Table 1. The nine types of graphs used for our experiment

The random graph generator we used labels the nodes numerically according to the order of their creation which, as such, makes task 1 amount to finding the greatest numeric label. Consequently, we decided to make this task more relevant by renaming the nodes alphabetically (from A to T on the twenty-node graphs, from A1 to F0 on the fifty-node graphs, and from A1 to K0 on the one-hundred-node graphs).

3.4.2 The population

The population that performed the evaluation consisted of post-graduate students and confirmed researchers in the fields of computer science. All the subjects knew what a graph was. No further knowledge of graph theory was required. The population consisted of 36 subjects, all of whom had previously seen a node-link representation of graphs. All the subjects participated voluntarily to the evaluation.

3.4.3 The evaluation program

We developed an evaluation program that represents the selected graphs according to both representation techniques. It then asks the user to perform the tasks and records the time to answer.

In terms of interaction, our program provides picking and selection mechanisms. On both visualizations, when the mouse goes over a node, it is highlighted in green as well as its incident links; nodes can also be selected through direct pointing, in which case they are highlighted in red as well as their incident links. Likewise, when the mouse goes over a link, it is highlighted in green as well as its endpoints. (Figure 1) These interactive enhancements were added to help users focus on graph elements after an initial testing showing a high level of frustration from users losing focus.

A demonstration made on a set of two graphs allowed us to explain how the representations should be read and how the various tasks could be performed. First, the instructor manipulated the system and provided guidelines. Then the user manipulated the system in order to make sure that the representations, the tasks and the interactions were well understood. At the end of the demonstration, we made sure that the user was ready to start the evaluation per se. When necessary, the instructor proposed to repeat the demonstration again. At the end, three instructions were given:

The user has to answer as quickly as possible.

The user has to answer correctly.

The user is allowed to move to the next question without answering before the answer time elapses in case he felt he was not able to answer.

To avoid memorization biases, the system selects a representation technique at random – matrix or node-link – and represents sequentially all nine graphs, asking the user to execute the seven tasks for each graph. Then, the system moves to the second technique and does the same. By interchanging the representation techniques, we make sure that the subjects had the same probability to start the evaluation with a series of visualizations belonging to either technique.

Each series was divided into two parts: the first included three simple graphs (graphs 1, 2 and 4) and allowed the user to get familiar with the system; the second included the six remaining graphs.

Furthermore, a learning effect was observed when a user was confronted to the matrix representation. We were able to measure such an undesirable effect in a series of ten preliminary tests where the system selected the graphs from the smallest to the largest and from the sparsest to the most connected. In spite of the increasing complexity of the displayed graphs, users would tend to answer more quickly as their understanding of matrix-based representations increased throughout the experiment. To level this effect, during the evaluation, our system selects the graphs at random within each half-series. In this way, the graphs have an equal probability to appear in the beginning, in the middle, or at the end of their respective half-series.

For tasks involving two nodes, (tasks 5, 6 and 7), the system selects both nodes beforehand in order to avoid spending time trying to locate them. Therefore, the time we measure corresponds exactly to the time for executing those tasks. Since each evaluation session contains a total of 126 questions (9 graphs x 2 visualization techniques x 7 tasks), we programmed three pauses: a ten-minute pause between the two series and a five-minute pause between the two halves of each series. Moreover, since the sessions are rather long, (a full hour of manipulation per user), we chose to limit the answer time to 45 seconds per question. When the time runs out, the system moves automatically to the next question and produces an audio feedback in order to notify the user. In this case, we consider that the representation is not effective for that task since the user was not able to provide the answer in the allotted time. The audio feedback also incites the user to hurry up for next questions.

3.4.4 Implementation and tuning

Node-link diagrams were laid out using AT&T's [1] open source graph layout program neato and the java drawing library grappa. We made our best effort to tune both representations in order to make the best use of them. We made the same interaction tech-

niques available on both visualizations. We paid attention to the size of nodes and the readability of their labels on node-link diagrams, however large or dense they got. We superimposed the labels of picked or selected nodes on a semi-transparent background, which eliminates the occlusion problems due to links overlapping over these nodes. Given that we are dealing with undirected graphs, we did not display any arrows at the extremities of the links, which significantly improves the node-link representation of dense graphs. Dealing with undirected graphs also simplifies the path lookup task on the matrix-based representation since links appear twice. We stored the parameters of the tuned node-link diagrams in the dot format and used those settings along the evaluation. We thus guarantee that the subjects are confronted with exactly the same representation for respectively all nine graphs. Likewise, we exploited the intrinsic orderability of the matrix representation and sorted its lines and columns alphabetically. The matrix being sorted instantaneously, the matrix-based visualizations did not require any preliminary tuning.

The evaluation was carried out on a Dell workstation having an NVIDIA GeForce2 accelerated video card, a dual Pentium III 1Ghz processor, 512 Mbytes of RAM, under Windows 2000. The display was performed in full screen mode on a 21" monitor. Subjects were seated at sixty centimeters from the monitor and executed all tasks using the mouse.

3.5 Results

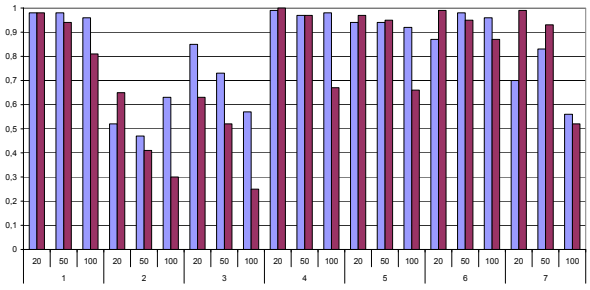


Figure 2 Percentage of correct answers split by task and by size. The matrix representation appears in blue and the node-link in purple.

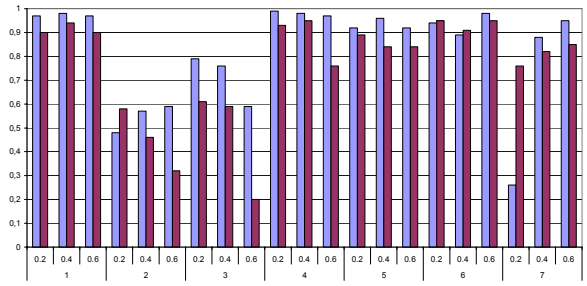


Figure 3 Percentage of correct answers split by task and by density. The matrix representation appears in blue and the node-link in purple.

The measurements were analyzed using a graphic qualitative method (Box-Plot) and a quantitative method (non parametric test of Wilcoxon). The latter makes it possible to compare the central position of two samples without prior knowledge of their distribution (normality for example). This test provides a p-value which is a probability. When the p-value is less than 5%, we conclude that the two samples have the same median value; otherwise, we conclude that the two samples have different median values.

On the following box-plot diagrams, blue boxes correspond to the matrix representation and purple boxes correspond to the node-link representation. On the y-coordinates, we represent the answer time. For each task, we display the evolution of time for the three graph sizes on diagrams labeled (a), and on the ones labeled (b) we display the evolution of time for the three chosen densities.

3.5.1 Estimation of the number of nodes (nodeCount)

On Figure 4a, on the matrix-based representation (x-coordinates 1, 3 and 5), median answer time and time distribution vary a little when size increases, whereas they grow notably on the node-link representation (x-coordinates 2, 4 and 6). We therefore conclude that with regard to this task, the readability of node-link diagrams deteriorates significantly when the size of the graph increases whereas the matrix-based representation is less affected by size. On Figure 4b, on the matrix-based representation (x-coordinates 1, 3 and 5), median answer time and time distribution increase a little when the density increases; they increase slightly on the node-link representation.

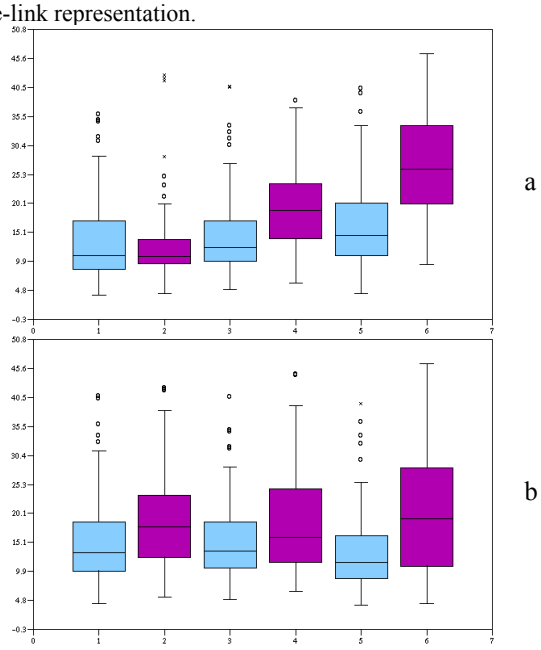


Figure 4 Distribution of answer time for "nodeCount" (a) split by size, (b) split by density

Moreover, in Figure 2, we note that, with regards to large graphs, 96% of the users have answered correctly using the matrix-based representation, against 81% using the node-link representation, that is to say a difference of 15%, deemed statistically significant according to Wilcoxon's test. On the matrix-based representation, the percentage of correct answers remains stable, around 97%, when density varies (Figure 3), which corresponds to a statistically significant improvement of 7% compared to the node-link representation for low and high densities.

The readability of the node-link representation is strongly affected when the number of nodes increases, and is slightly affected when link density increases, whereas the readability of matrix-based representations is slightly affected by size variation and is not sensitive to density variation. On top of that, using the matrix-based representation, users answer faster when the size and density are medium or large.

3.5.2 Estimation of the number of links (linkCount)

Estimating the number of links in the graph seems relatively independent of size or link density when these parameters take medium or large values. On Figure 5a (x-coordinates 2 and 4), there is a gap in answer time between small and medium-sized graphs and, on Figure 5b (x-coordinates 2 and 4), between sparse and moderately dense graphs. However, there seems to be no difference between the two techniques for any given size or density. On Figure 2, the matrix-based representation records 57% of correct answers on large graphs. This figure goes as low as 25% using the node-link representation, that is to say a significant discrepancy of 27 % compared to matrices. The difference recorded with regard to small and medium-sized graphs respectively in favor of the node-link representation and the matrix-based representation is statistically insignificant. Similar conclusions can be drawn with regard to link density (Figure 3).

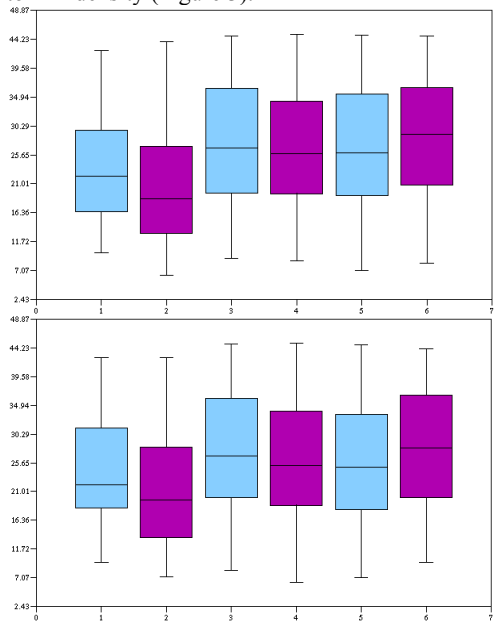


Figure 5 Distribution of answer time for "linkCount" (a) split by size, (b) split by density

3.5.3 Finding the most connected node (mostConnected)

When achieving this task, we note that, with regard to answer time, both techniques are sensitive when size increases (Figure 6a), whereas they are slightly affected when link density increases (Figure 6b). We cannot differentiate these methods with regard to this task based on answer time only.

Nevertheless, according to Figure 2, 85% of the users execute this task correctly on small graphs using a matrix-based representation against 63% of correct answers with the node-link representation. On medium-sized graphs, we have 73% of correct answers using a matrix against 52% using node-links. Lastly, on large graphs, we record 57% of correct answers using a matrix against only 25% of correct answers with node-link diagrams. These differences are deemed statistically significant using Wilcoxon's test. Similar conclusions can be reached with regard to density on Figure 3.

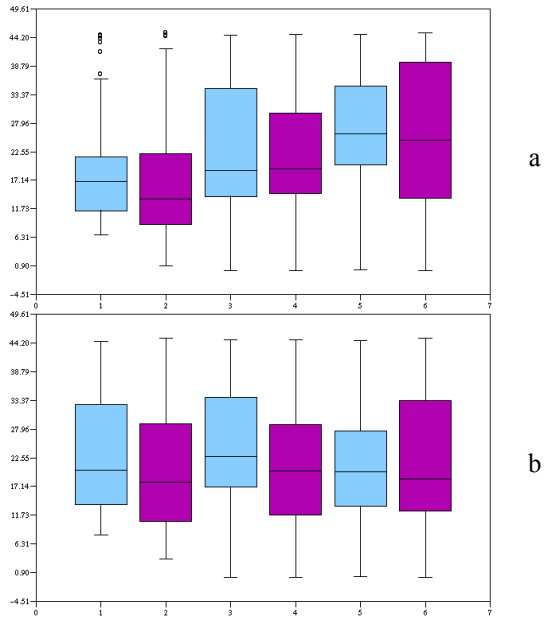


Figure 6 Distribution of answer time for "mostConnected" (a) split by size, (b) split by density

3.5.4 Finding a specified node (findNode)

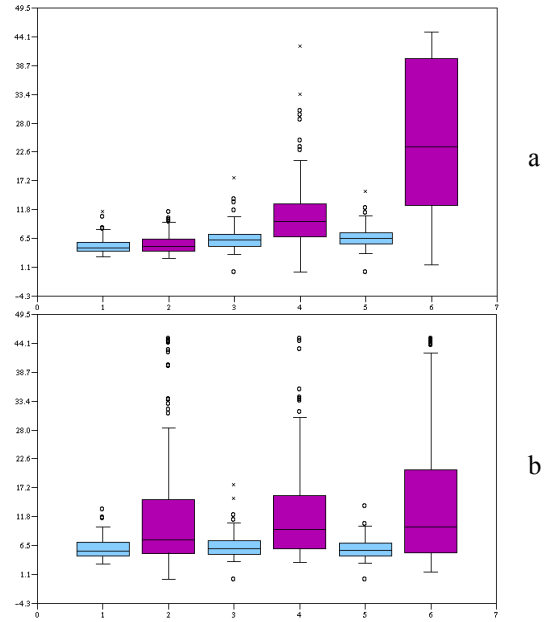


Figure 7 Distribution of answer time for "findNode" (a) split by size, (b) split by density

When considering answer time, we can see that the readability of node-link diagrams deteriorate quickly when the size of the graph increases (Figure 7a) and are moderately affected by link density (Figure 7b), whereas the answer time on the matrix-based representation deviates a little when the size increases and does not seem to be affected at all by link density. The dispersion of answer time is very small using matrix-based representation in both cases. When dealing with small graphs, both representations perform equally well. The percentage of correct answers (Figure 2) is high, almost 98%, using the matrix-based representation, irrespective of the size of the graph. The percentage of correct answers is

equally good using node-link diagrams, except for large graphs whose score falls as low as 67%, with a discrepancy of 31% compared to matrix-based representations. Similar conclusions can be drawn when link density varies (Figure 3).

3.5.5 Finding a link between two nodes (findLink)

Using the node-link representation, the larger the graph the longer it takes to look up a link in the graph (Figure 8a); the answer time does not vary significantly when link density increases (Figure 8b). Using matrices, this task is insensitive to size and density variation. For large graphs, and for medium and high link density, a significant gap is measured in favor of matrix-based representations. A significant difference is measured in favor of node-link diagrams for small graphs. Both representations record excellent percentages of correct answers, about 95%, for small and medium-sized graphs (Figure 2). For large graphs, the matrix-based representation records 92% of correct answers against 66% with the node-link diagrams, that is a discrepancy of 26%.

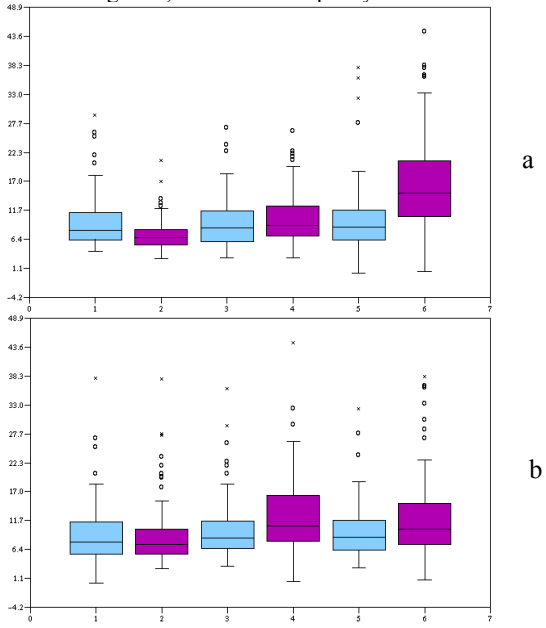


Figure 8 Distribution of answer time for "findLink" (a) split by size, (b) split by density

3.5.6 Finding of a common neighbor (findNeighbor)

Using the node-link representation, the larger the graph the longer it takes to say whether a common neighbor exists (Figure 9a); the median answer time is marginally affected when link density increases (Figure 9b). On the matrix-based representation, size variation has no impact on answer time, while median answer time and value dispersion improve slightly when link density is large.

When dealing with small graphs, node-link diagrams record 99% of correct answers with a lead of 12% over matrices. Matrix-based representations take an equivalent lead when dealing with large graphs, towering at 96% of correct answers. Both techniques record a similar percentage of correct answers on medium-sized graphs (Figure 2). When link density varies (Figure 3), both representations score about 90% of correct answers.

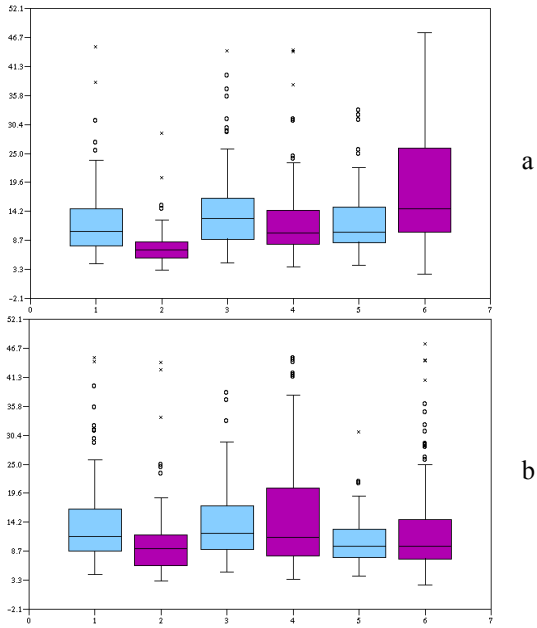


Figure 9 Distribution of answer time for "findNeighbor" (a) split by size, (b) split by density

3.5.7 Finding a path between two nodes (findPath)

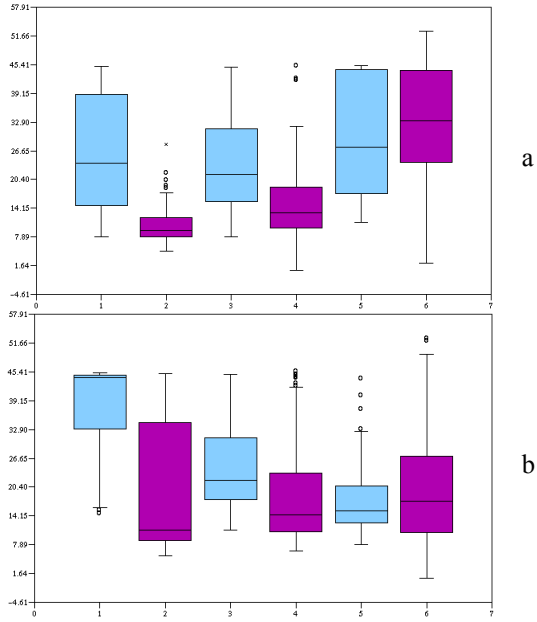


Figure 10 Distribution of answer time for "findPath" (a) split by size, (b) split by density

Finding a path between two nodes proves to be increasingly difficult using node-link diagrams when the size increases (Figure 10a), whereas the median answer time increases slightly when link density increases (Figure 10b). The matrix-based representation is much worse for this task except for very dense graphs where it outperforms the node-link representation (Figure 10b). This fact is confirmed by the percentage of correct answers which towers at 95% for the matrix-based representation against 85% for node-links when dealing with large link density (Figure 3). On small graphs, 99% of the users answer correctly using node-link diagrams against 70% only using matrices. On medium-sized

graphs, node-link diagrams record 93% of correct answers with a lead of 10% over matrices. For large graphs, every other user answers correctly using both representations with a statistically insignificant lead in favor of matrices. Lastly, this task is clearly in favor of node-link diagrams when visualizing sparse graphs.

4 Discussion

We expected the readability of node-link diagrams to deteriorate when the size of the graph and its link density increase. This hypothesis was confirmed for the seven tasks we selected. Only for "findPath" task did node-link diagrams prove superior to matrix-based representations, although their performance deteriorates on large and dense graphs. This conclusion must however be qualified since this task is difficult to carry out visually when the distance between the extremities is greater than two or three arcs, as shown in [13].

Another hypothesis concerned the impact of orderability of matrices on node and link finding tasks. "findNode" and "findLink" tasks validate this hypothesis for large graphs and for dense graphs.

As far as "linkCount" task is concerned, both visualizations record a large share of erroneous answers. We account for that – but this has yet to be proven through experimentation – that the number of links is intrinsically difficult to estimate on node-link diagrams and that users failed to compute it correctly using matrices. Indeed, links are displayed twice because we considered undirected graphs in our study.

We may also question the extensibility of the results obtained in this evaluation to other node-link layout programs than the one we chose in our experimentation. However, based on earlier works [11], we can safely assert that, with regard to small graphs, the layout program has very little impact on the readability of the displayed output and would not change the trends we observed.

We may further highlight that all the users who took part in the experiment were familiar with node-link diagrams whereas none had previously heard about the matrix-based representation of graphs. Since they were given little training (first, users would watch the instructor perform the tasks on a graph, and then they would train on one similar graph), we expect users familiar with both representations to perform even better with matrices.

As a first approach, we have split the data by size and by density in order to measure the effect of these variables on the readability of graph representations. To this end, we have done our best effort to isolate those factors and make sure that no other considerations would interfere with the tasks. However, further analysis of the results is required in order to check for a combined effect of size and density of graphs on the readability of their representations. It would also be meaningful to break down the results for each of the nine graphs we studied for a better control over the parameters and a finer understanding of the results. For instance, finding a path on a node-link diagram representing a sparse graph proves all the more difficult than the shortest path between the extremities is long. In this case, the density of the graph may be a misleading indicator; the length of the shortest path may be a better choice and should be taken into account. Conversely, in dense graphs, the shortest path is likely to be one or two links long, but the visual clutter produced by links on node-link diagrams makes this task unfeasible, while matrices perform very well.

In this evaluation, we compare two representations of graphs, a matrix-based representation and a node-link representation produced by a force directed algorithm, against nine random graphs and a set of seven exploration tasks. In this context, the recommendations we can derive from this study are: for small graphs, node-link diagrams are always more readable and more familiar than matrices. For larger graphs, the performance of node-link diagrams deteriorates quickly while matrices remain readable with a lead of 30% of correct answers, with comparable if not better answer time. For more complex tasks such as "findPath", we are convinced that an appropriate interaction is always preferable, for example by selecting a node and displaying all the possible paths starting from it and ending at a pointed node. On the matrix-based representation, this path can be displayed using curves connecting adjacent links, i.e. connecting the cells representing those links.

5 Conclusion

In this paper, we have listed generic tasks for the visualization of graphs and have compared two representations of graphs on a subset of these tasks. We have proved theses techniques to be complementary: node-link diagrams are well suited for small graphs, and matrices are suitable to large or dense graphs. Path related tasks remain difficult on both representations and require an appropriate interaction that helps perform them.

The matrix-based representation seems therefore under exploited nowadays, despite its quick layout and its superior readability with regard to many tasks. We think that a wider use of this representation will result in a greater familiarity and will consequently improve its readability. We currently use the matrix-based representation for the real-time monitoring of constraint-oriented programs where graphs evolve dynamically, both in size and activity. The results we are obtaining are quite encouraging. We are investigating clustering and aggregation techniques of matrices for the visualization of very large graphs, about tens of thousands vertices.

6 Acknowledgements

We would like to thank Pierre Dragicevic, Véronique Libérati and Vanessa Tico for their time and advice. We are grateful to the users who volunteered and took part in this evaluation and to the members of the RNTL OADYMPPAC project.

References

[1] AT&T Labs Research. Graphviz - open source graph drawing software, 2004.

[2] http://www.research.att.com/sw/tools/graphviz/

[3] Battista, G.D., Eades, P., Tamassia, R. and Tollis, I.G. Graph Drawing. Prentice Hall, 1999.

[4] Becker, R.A., Eick, S.G. and Wills, G.J. Visualizing network data. IEEE Transaction on Visualizations and Graphics, 1 (1). 16-28.

[5] Bertin, J. Sémiologie graphique : Les diagrammes - Les réseaux - Les cartes. Editions de l'Ecole des Hautes Etudes en Sciences, Paris, France, 1967.

[6] Cohen, R.F., Eades, P., Lin, T. and Ruskey, F., Volume upper bounds for 3D graph drawing. in Proceedings of the 1994 conference of the Centre for Advanced Studies on Collaborative research, (Toronto, Ontario, Canada, 1994), IBM Press.

[7] Ghoniem, M., Jussien, N. and Fekete, J.-D., VISEXP: visualizing constraint solver dynamics using explanations. in FLAIRS'04: Seventeenth international Florida Artificial Intelligence Research Society conference, (Miami Beach, FL, 2004), AAAI press.

[8] Ham, F.v., Using Multilevel Call Matrices in Large Software Projects. in Proc. IEEE Symp. Information Visualization 2003, (Seattle, WA, USA, 2003), IEEE Press, 227-232.

[9] Herman, I., Melançon, G. and Marshall, M.S. Graph Visualization and Navigation in Information Visualization: a Survey. IEEE Transactions on Visualization and Computer Graphics, 6 (1). 24-43.

[10] ISPT, Waseda University. Random Graph Server. http://www.ispt.waseda.ac.jp/rgs/index.html

[11] Purchase, H.C., The Effects of Graph Layout. in Proceedings of the Australasian Conference on Computer Human Interaction, 1998, p. 80.

[12] Purchase, H.C., Carrington, D.A. and Allder, J.-A. Empirical Evaluation of Aesthetics-based Graph Layout. Empirical Software Engineering, 7 (3). 233-255.

[13] Purchase, H.C., Cohen, R.F. and James, M.I. An Experimental Study of the Basis for Graph Drawing Algorithms. The ACM Journal of Experimental Algorithmic, Volume 2, Article 4, 1997.

[14] Ware, C., Purchase, H.C., Colpoys, L. and McGill, M. Cognitive measurements of graph aesthetics. Information Visualization, 1 (2). 103-110.