Information Visualization (2005) 4, 114–135

www.palgrave-journals.com/ivs

On the readability of graphs using node-link and matrix-based representations: a controlled experiment and statistical analysis

Mohammad Ghoniem[1]
Jean-Daniel Fekete[2]
Philippe Castagliola[3]

[1]Ecole des Mines de Nantes, Nantes, Cedex 3, France; [2]INRIA Futurs/LRI, Bât 490, Université Paris-Sud, Orsay, Cedex, France; [3]IRCCyN/IUT de Nantes, Rue Christian Pauc – La Chantrerie, Nantes, Cedex 3, France.

Correspondence: Mohammad Ghoniem, Ecole des Mines de Nantes, 4 rue Alfred Kastler. B.P.20722 44307 NANTES Cedex 3, France.
Tel: +33 2 97 54 35 05
Fax: +33 2 51 85 82 49
E-mail: Mohammad.Ghoniem@emn.fr

Received: 30 October 2004
Accepted: 1 February 2005
Online publication date: 19 May 2005

Abstract
In this article, we describe a taxonomy of generic graph related tasks along with a computer-based evaluation designed to assess the readability of two representations of graphs: matrix-based representations and node-link diagrams. This evaluation encompasses seven generic tasks and leads to insightful recommendations for the representation of graphs according to their size and density. Typically, we show that when graphs are bigger than twenty vertices, the matrix-based visualization outperforms node-link diagrams on most tasks. Only path finding is consistently in favor of node-link diagrams throughout the evaluation.
Information Visualization (2005) **4,** 114–135. doi:10.1057/palgrave.ivs.9500092

Keywords: Visualization of graphs; adjacency matrices; node-link representation; readability; evaluation

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.[1] Most works strive to optimize algorithms complying with such rules, but they rarely try and validate them from a cognitive point of view. Recently, Purchase *et al*.[2] tackled this problem through on-paper and online experiments.[3,4] These works involved small handcrafted graphs (graphs with 20 vertices and 20–30 edges), five aesthetic criteria and eight graph layout algorithms. The authors point out that while some aesthetic criteria taken separately may improve the perception of the graph at hand, it is difficult to claim that an algorithm brings about such an improvement. Moreover, Ware and Purchase[5] set up a study aimed 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. In the Information Visualization (Infovis) community, many node-link variants have been experimented, both in 2D and 3D.[6,7] However, as soon as the size of the graph or the link density increases, all these techniques face 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

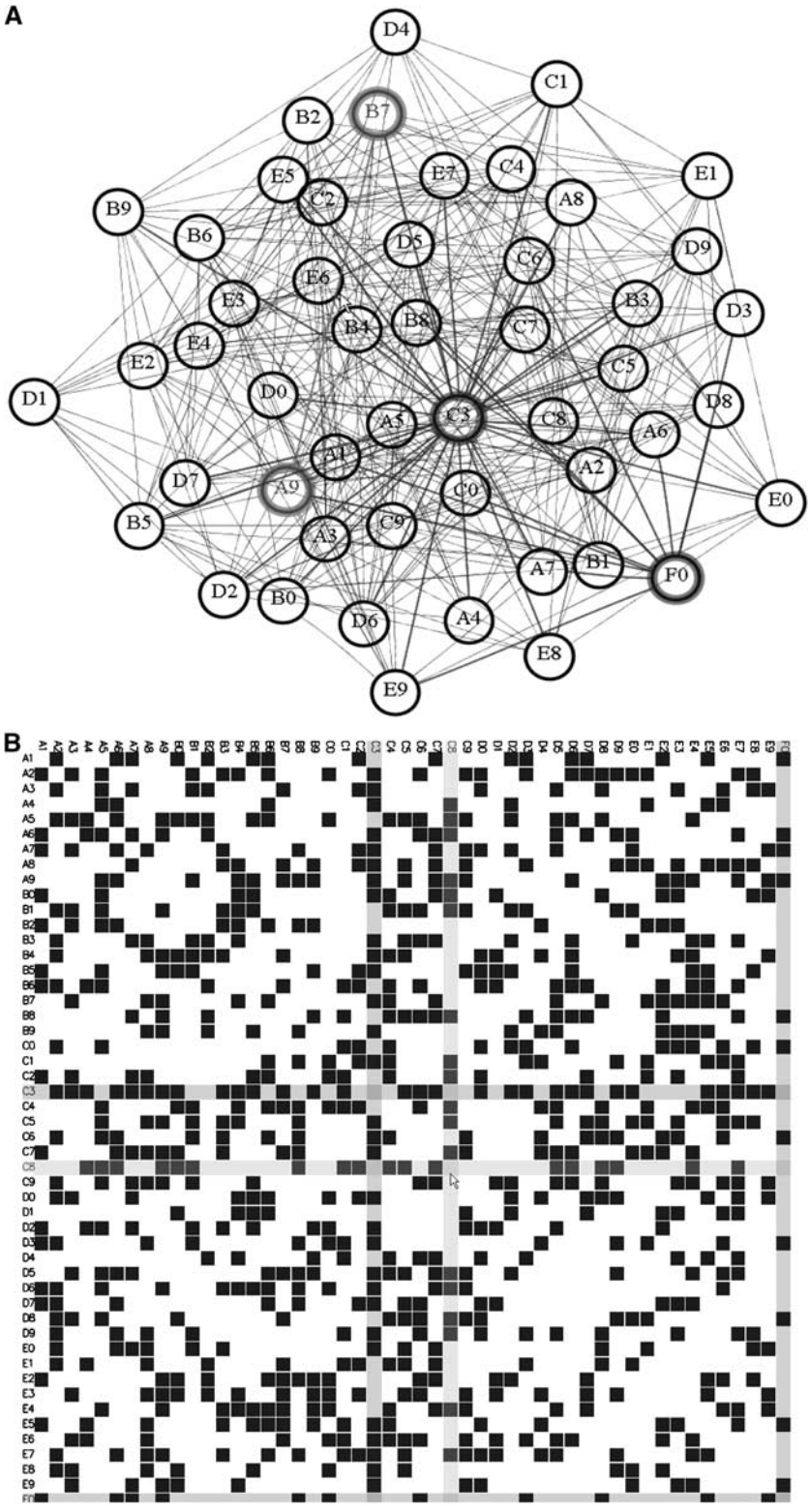

Figure 1 Two visualizations of the same undirected graph containing 50 vertices and 400 edges. The node-link diagram (A) is computed using the 'neato' program and the matrix representation (B) is computed using our VisAdj program.

comparing the two representations in order to show their respective advantages with regard to a set of generic analysis tasks. We elaborate on our previous work[8] presented at the infovis symposium and provide additional statistics via multiple regression analysis along with a more in-depth interpretation of the collected results.

The matrix-based visualization of graphs

The matrix-based visualization of graphs (Figure 1B) relies from a formal standpoint on the potential representation of a graph via its boolean-valued connectivity matrix where rows and columns represent the vertices of the graph. Although conventions may vary for directed graphs, columns and rows generally represent the origin of edges and their end point vertices, respectively. When two vertices are connected, the cell at the intersection of the corresponding row and column contains the value 'true'. Otherwise, it takes on the value 'false'. Boolean values may be replaced with valued attributes associated with the edges that can provide a more informative visualization.

The matrix-based representation of graphs offers an interesting alternative to the traditional node-link diagrams. In,[9] Bertin shows that it is possible to reveal the underlying structure of a network represented by a matrix through successive permutations of its rows and columns. In,[10] the authors visualize the load distribution of a telecommunication network by using a matrix but their effort aims mostly 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,[11] the authors implemented a multi-scale matrix-based visualization that represents the call graph between software components in a large medical imagery application. In,[12] we have shown that a matrix-based representation can be used to effectively grasp the structure of a co-activity graph and to 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.

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 seems potentially endless. For instance, this fact can be easily illustrated by constructing a Website-related graph and the associated list of tasks that one can carry out or wish to carry out on such a graph. However, for tractability reasons, we tackled the problem by focusing on a set of 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 user's ability to answer some relevant questions about the overview. As far as graphs are concerned, some questions may bear on topological considerations or topology-related attributes. The most generic questions related to the topology of a graph – that is, the ones independent of the semantics of data – are related to its vertices, links, paths and sub-graphs. We extracted from the 'Glossary of graph theory' entry of the Wikipedia Free Encyclopedia a set of important concepts and organized basic tasks related to them as follows:

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: They include 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.

Readability of a graph representation

The readability of a graphic representation can be defined 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 considered very readable for the task. If, however, 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 specific graph instances at hand, their familiarity to users, their meaning, and the layout used to visualize them. In our evaluation, we only compare random graphs which are meaningless and equally unfamiliar to users and hence, we only focus on abstract characteristics of graphs. We choose a popular graph layout program called 'neato', part of the GraphViz[13] package to compute the node-link diagrams. It could be argued that an alternative layout program might provide a more readable layout according to our tasks. This is certainly true for actual figures but we believe that the overall trends would be similar when the size and density of graphs increase.

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 greatly influence the readability of this representation. We define the link density d in a graph as $d = \sqrt{l/n^2}$, 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. Although topologically meaningful, this definition is not scale invariant since the number of potential edges increases in the square of the number of vertices.

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 representations, unless nodes become cluttered on node-link diagrams. Counting links should be easier on matrices since no occlusion interferes with the task. Also, finding the most connected node should perform better on matrices for dense graphs. In fact, in the context of 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 row 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 for small-sized graphs.

Experimental setup

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.[14] 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.

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 50-node graphs, and from A1 to K0 on the 100-node graphs).

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.

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

Table 1 The nine types of graphs used for our experiment

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

elements after an initial testing showing a high level of frustration from users losing focus.

A demonstration made on a set of two graphs exposed the user to the proper reading of representations and the various tasks to 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*. The instructor proposed to repeat the demonstration as necessary. At the end of this introductory demonstration, three instructions were given:

1. The user has to answer as quickly as possible.
2. The user has to answer correctly.
3. The user is allowed to move to the next question without answering before the answer time elapses in case he feels he is 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 proceeds in the same fashion. 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 10 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, (findLink, findNeighbor and findPath), 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 $\times 2$ visualization techniques $\times 7$ tasks), we programmed three pauses: a 10-min pause between the two series and a 5-min 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 s 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.

Implementation and tuning Node-link diagrams were laid out using AT&T's[13] 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 techniques 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 end points of the links, which significantly improves the node-link representation of dense graphs. Dealing with undirected graphs would also simplify 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 rows 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 1 Ghz processor, 512 Mbytes of RAM, under Windows 2000. The display was performed in full screen mode on a 21″ monitor. Subjects were seated at 60 cm from the monitor and executed all tasks using the mouse.

Results

Throughout this paper, we plotted in light gray the statistics corresponding to matrices and in dark gray those related to node-link diagrams. The measurements were analyzed with 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 central values.

On the box-plot diagrams displayed subsequently, we represent the answer time on the y-axis. 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.

In order to fully understand the effect of both variables – size and density – and reveal any possible interaction between them, we ran a multiple regression analysis[15] on the collected measurements. We assume that the response y of a system is supposed to depend on $k=2$ controllable input variables (ξ_1, ξ_2) called natural variables. In our case, ξ_1 and ξ_2 are, respectively, the density and the size of the graphs. The true response function $y=f(\xi_1, \xi_2)$ is supposedly unknown. The main goal of the *Multiple Regression Analysis* is to approximate the unknown true response function with a 'simple' known function like:

$$y = a_0 + a_1\xi_1 + a_2\xi_2 + a_{12}\xi_1\xi_2, \quad (1)$$

where a_0, a_1, a_2, a_{12} are the $p=k+2=4$ regression coefficients. This model is called a linear model plus interactions model. In order to estimate the regression coefficients, we have to perform $n>p$ experiments where the natural variables (ξ_1, ξ_2) take different predefined values. In our case, the predefined values are $\{0.2, 0.4, 0.6\}$ for ξ_1 and $\{20, 50, 100\}$ for ξ_2. Let $\mathbf{X}$ and $\mathbf{y}$ the matrix and vector defined as:

$$\mathbf{X} = \begin{pmatrix} 1 & \xi_{11} & \xi_{12} & \xi_{11}\xi_{12} \\ \vdots & \vdots & \vdots & \vdots \\ 1 & \xi_{n1} & \xi_{n2} & \xi_{n1}\xi_{n2} \end{pmatrix} \mathbf{y} = \begin{pmatrix} y_1 \\ \vdots \\ y_n \end{pmatrix}, \quad (2)$$

where (ξ_{i1}, ξ_{i2}) are the predefined values chosen during the ith experiment and where y_i is the corresponding response. An estimation of the regression vector $\mathbf{a}=(a_0, a_1, a_2, a_{12})^{\mathrm{T}}$ can be obtained using the following relation:

$$\hat{\mathbf{a}} = (\mathbf{X}^T\mathbf{X})^{-1}\mathbf{X}^T\mathbf{y}. \quad (3)$$

The significance of each regression coefficients can be evaluated with the computation of a *P*-value (not presented here). If the *P*-value is less than 0.05 then the coefficient is deemed non null, and the corresponding variable (density or size) is deemed influent on the response. On the other side, if the *P*-value is larger than 0.05 then the coefficient is considered to be null and the corresponding variable is supposed to have no influence on the response.

Finally, we obtain a model represented by a surface in a 3D space whose axes are respectively the two variables (size and density in our case) and the response (answer time in our case). It may also be mentioned that the 3D graphs take into account statistically insignificant terms that were left out for the sake of readability in the corresponding equations. Moreover, in order to help figure out the shape of the 3D model, we display the projection of the model on plan XY; the lines plotted in the XY plan correspond to the projection of isovalue contours in the 3D surface. Lastly, we may highlight once again that with regards to tasks involving two nodes such as findLink, findNeighbor and findPath, we made sure that the user did not spend any time finding the nodes in the question by automatically highlighting them. Hence, the measurements faithfully reflected the time spent at these tasks.

In the sequel, we provide a detailed account of the results we obtained. In brief, one may say that, except for findPath, matrix-based visualizations outperform node-link diagrams with medium to large graphs and with dense graphs.

Estimation of the number of nodes (nodeCount) In Figure 4A, on the matrix-based representation (the light gray boxes), median answer time and time distribution vary a little when size increases, whereas they grow notably on the node-link representation (the dark gray boxes). 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. In Figure 4B, on the matrix-based representation, median answer time and time distribution increase a little when the density increases; they increase slightly on the node-link representation.

Moreover, in Figure 2, we note that, with regard 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.

In Figure 5, we display the regression models of answer time with regard to the 'nodeCount' task. In Figure 4A, answer time using a matrix (T_{MX}) is modeled by the following regression equation:

$$T_{\mathrm{MX}} = 18.8999 - 15.3938\times d + 0.157116\times d\times s, \quad (4)$$

where d stands for density and s for size. In Figure 4B, answer time using a node-link diagram (T_{NL}) for the same task is modeled by the following regression equation:

$$T_{\mathrm{NL}} = 13.7151 - 10.6864\times d + 0.302776\times d\times s. \quad (5)$$

Eq. (4) shows some interaction between size and density of graphs (see the last coefficient in the equation) and a favorable effect of density on matrices (the coefficient related to density is negative). Likewise, Eq. (5) shows strong interaction between size and density with node-links (see the highest value at (0.6, 100)) and a favorable effect of density on answer time. Size did not seem to play a statistically significant role in the estimation of node

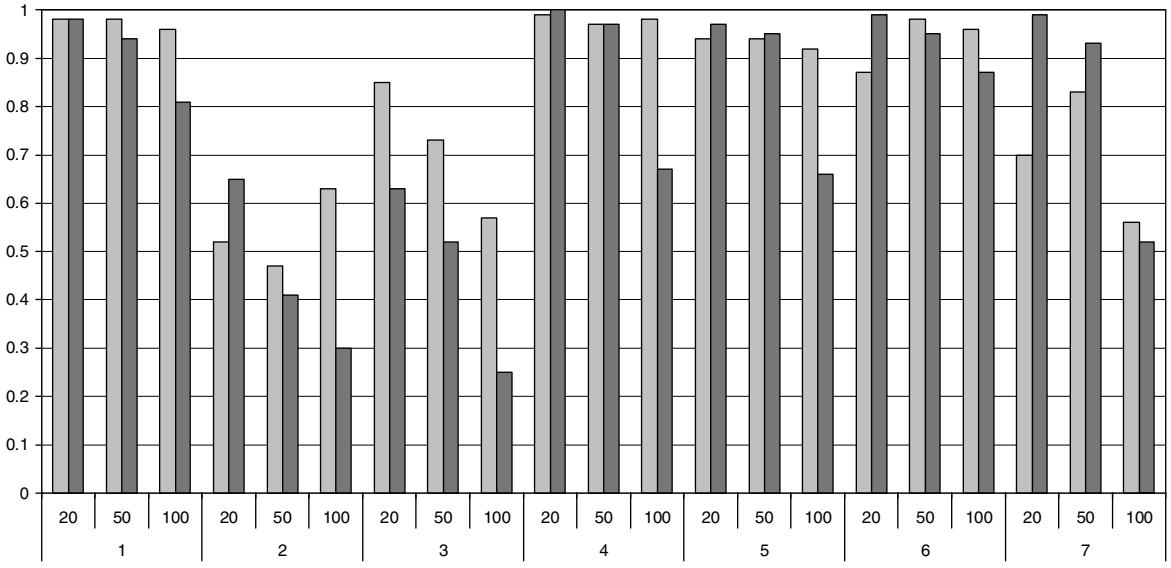

Figure 2 Percentage of correct answers by task and by size. Matrices appear in light gray, node-links in dark gray.

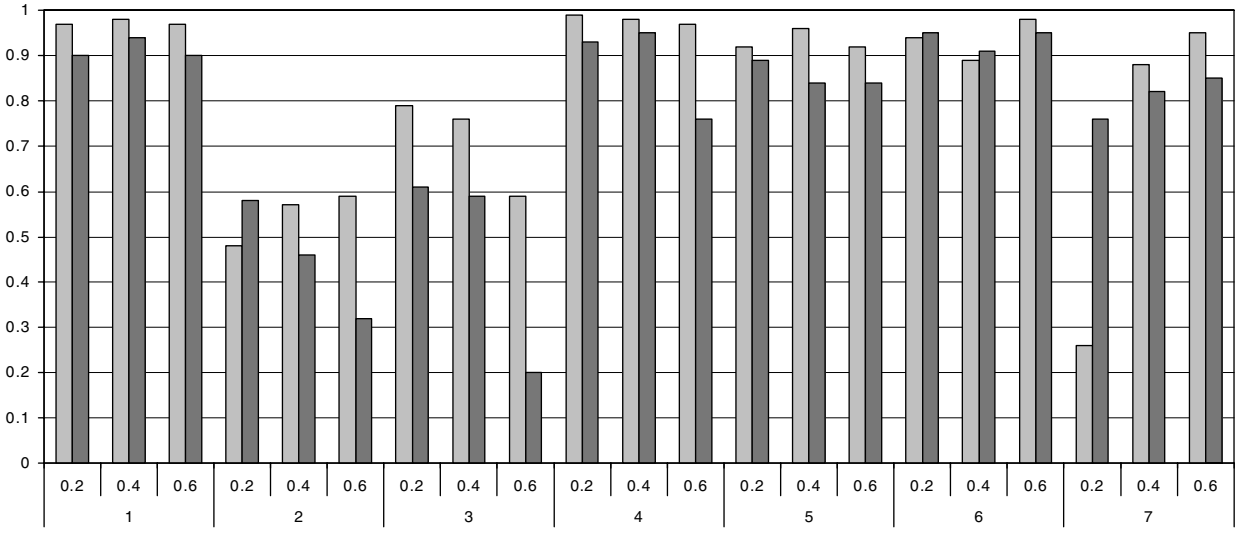

Figure 3 Percentage of correct answers by task and by density. Matrices appear in light gray; node-links in dark gray.

count in either case. This can be explained by the fact that users really estimated the size of graphs without resorting to an exact count; the complexity of many node-link diagrams would deter them if they tried! This is also a likely explanation of the favorable effect of density.

Figure 5 and the related equations suggest that the readability of the node-link diagrams is strongly affected by an interaction between density and size, while size has no significant independent effect. The readability of matrices is less influenced by the interaction between size and density – although such interaction is observed – and is not sensitive to size variation. On top of that, matrices allow carrying out this task more quickly when size and density are medium or large. Indeed, the dark gray surface takes off for these values while the light gray surface remains unchanged.

Estimation of the number of links (linkCount) Based on Figure 6, the estimation of the number of links in the graph seems relatively independent of size or link density when these variables take medium or large values. On Figure 6A (*x*-coordinates 2 and 4), there is a gap in answer time between small and medium-sized graphs and, on Figure 6B (*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

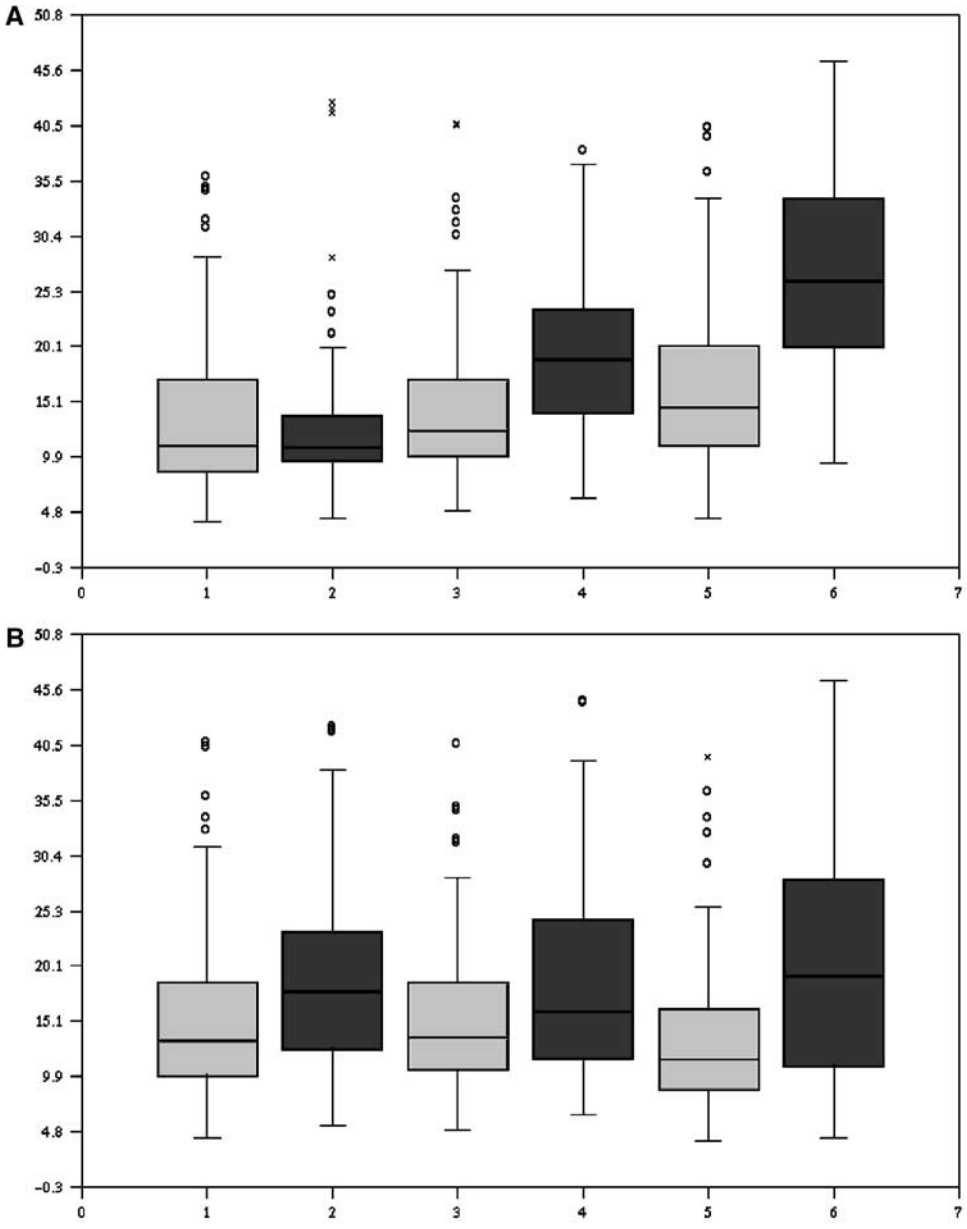

Figure 4 Distribution of answer time for 'nodeCount' (A) split by size, (B) split by density.

or density. In 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 for link density (Figure 3).

The regression equations related to Figure 7 are respectively as follows:

$$T_{MX} = 19.6512 + 0.0844522 \times s, \tag{6}$$

$$T_{NL} = 38.3796 \times d + 0.246053 \times s - 0.374642 \times d \times s. \tag{7}$$

It appears that answer time using matrices is completely independent of density, whereas answer time using node-link diagrams depends on both size and density and is favorably impacted by an interaction effect between both variables. While node-links stand out clearly when it comes to small graphs (Figures 6 and 7), they take off quickly and provide answer times quite similar to those obtained with matrices. Only the ratio of correct answers makes a difference between the two representations in favor of matrices.

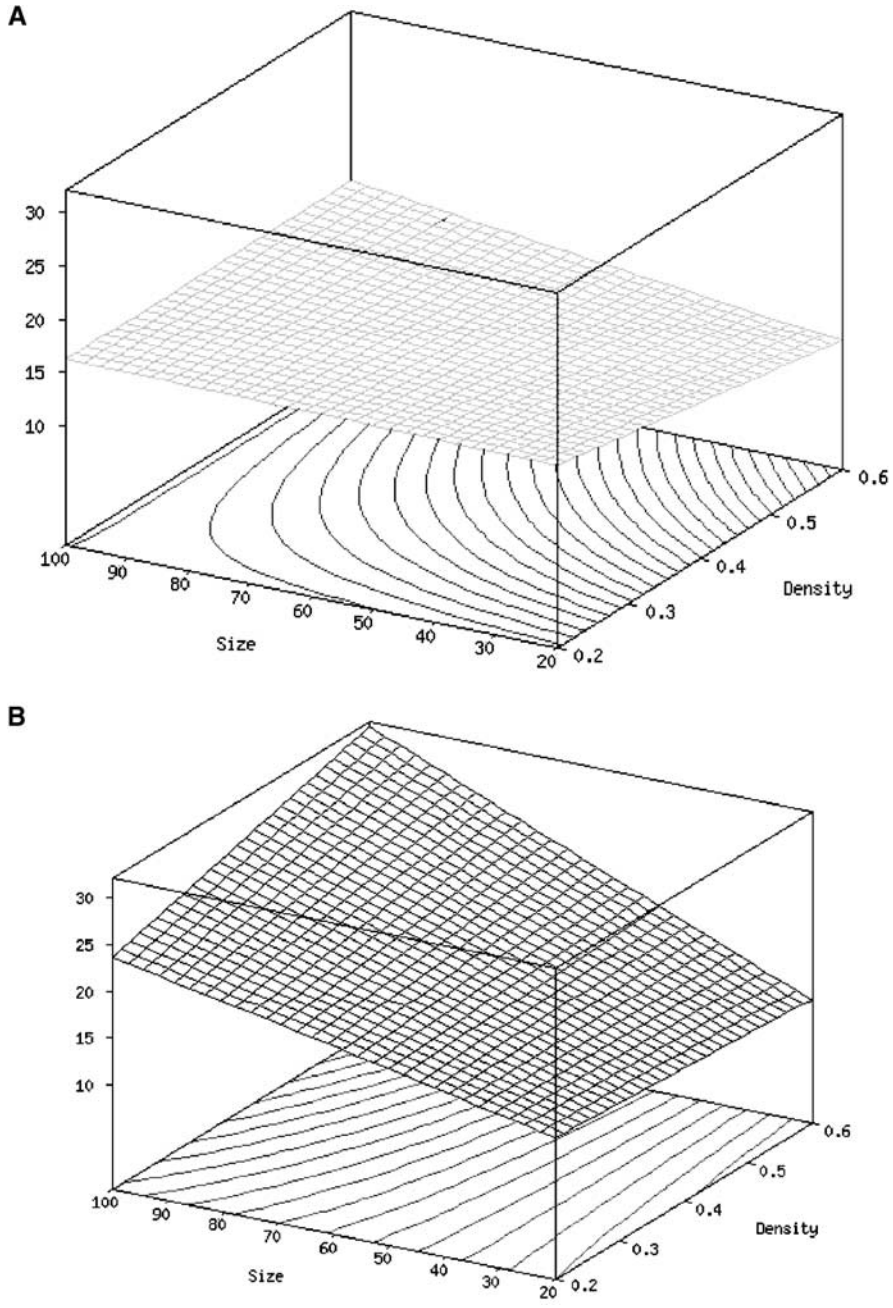

Figure 5 Model of response time with regard to nodeCount for (A) matrices and (B) node-links.

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 8A), whereas they are slightly affected when link density increases (Figure 8B). 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.

The regression equations related to Figure 9 are:

$$T_{\mathrm{NL}} = 47.0504 \times d + 0.406933 \times s - 0.757307 \times d \times s, \quad (8)$$

$$T_{\mathrm{MX}} = 14.3773 + 0.179544 \times s. \quad (9)$$

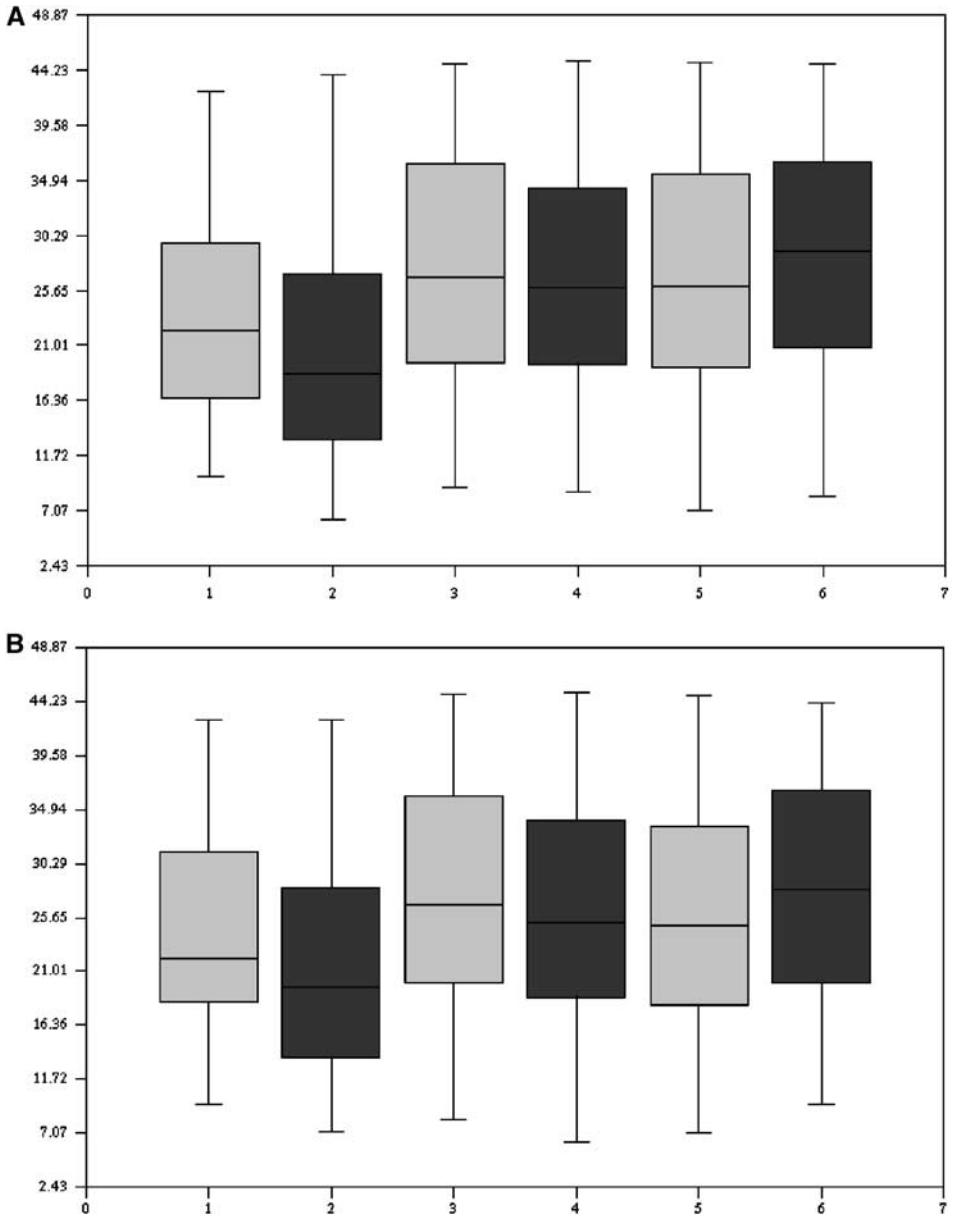

Figure 6 Distribution of answer time for 'linkCount' (A) split by size, (B) split by density.

Matrices prove to be completely independent of density; the answer time required for finding the most connected vertex depends on size only with a slope of 18%. As for node-links, answer time depends very much on both variables, despite a strong interaction between both variables reduces the measured values of answer time. (See how the surface bends at the middle on Figure 9B.) Bearing the large ratio of incorrect answers in mind (Figure 2), it seems that some users would randomly select any answer to this question when the graphs became too complex, a behavior that we had the opportunity to see while watching the users.

Finding a specified node (findNode) When considering answer time, we can see that the readability of node-link diagrams deteriorates quickly when the size of the graph increases (Figure 10A) and are moderately affected by link density (Figure 10B). In contrast, 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

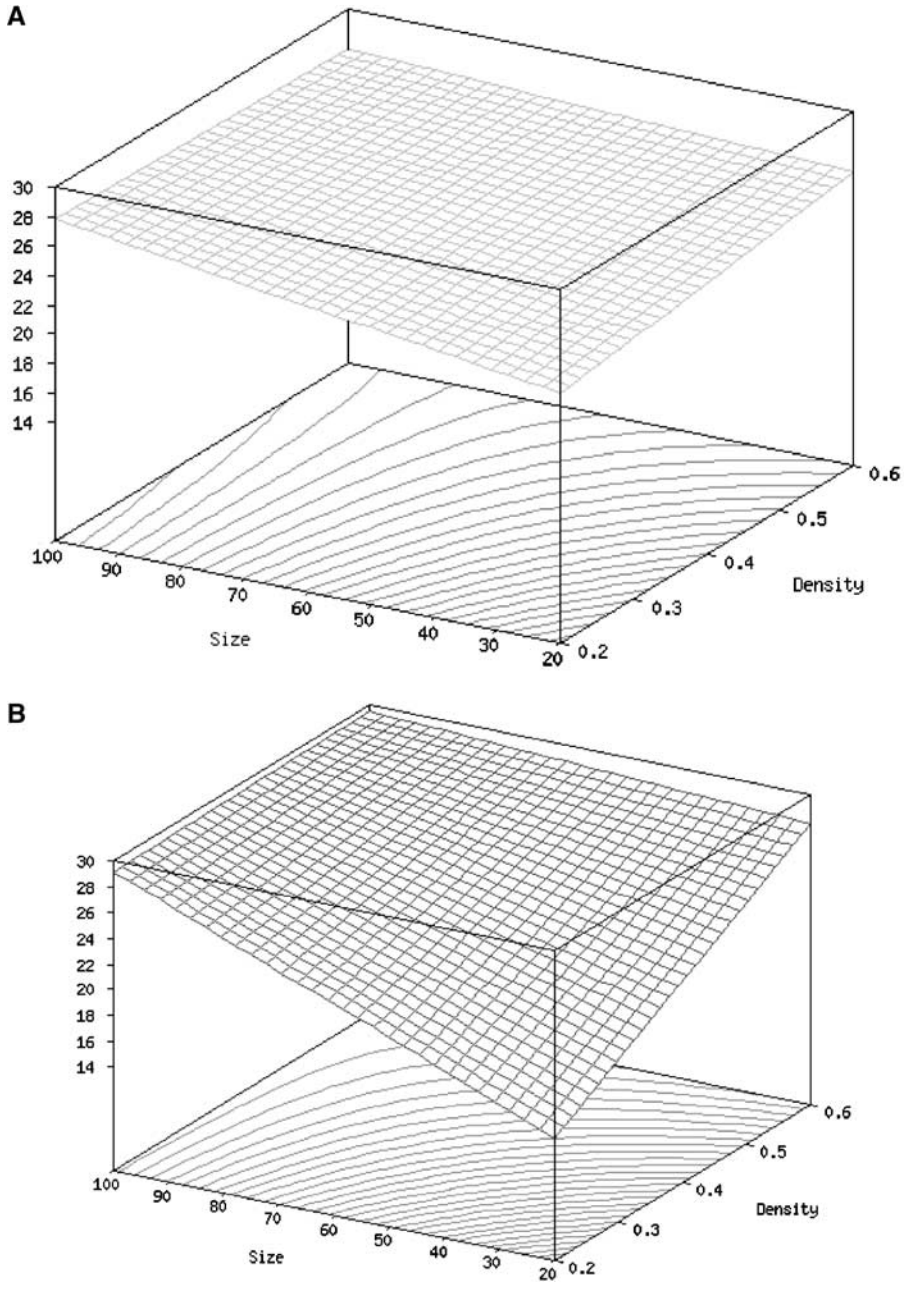

Figure 7 Model of response time with regard to linkCount for (A) matrices and (B) node-links.

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).

The regression equations related to Figure 11 are as follows:

$$T_{\mathrm{MX}} = 6.02597, \tag{10}$$

$$T_{\mathrm{NL}} = 0.179424 \times s + 0.179913 \times d \times s. \tag{11}$$

Eq. (10) confirms what we already know, more precisely the fact that looking up a vertex in an ordered representation (the matrix) does not depend on size or density of graphs. Node-link diagrams seem independent of density; this representation suffers however from the size of graphs (i.e. the number of nodes therein) and from an interaction between size and density (see the dark gray surface takes off until it reaches its maximum value at the far top corner). This may be accounted for by the complexity induced by the unpredictable layout of nodes in node-link diagrams, whatever the density may be. Size proves to be more preponderant in this case since it increases substantially the list of nodes to be scanned through.

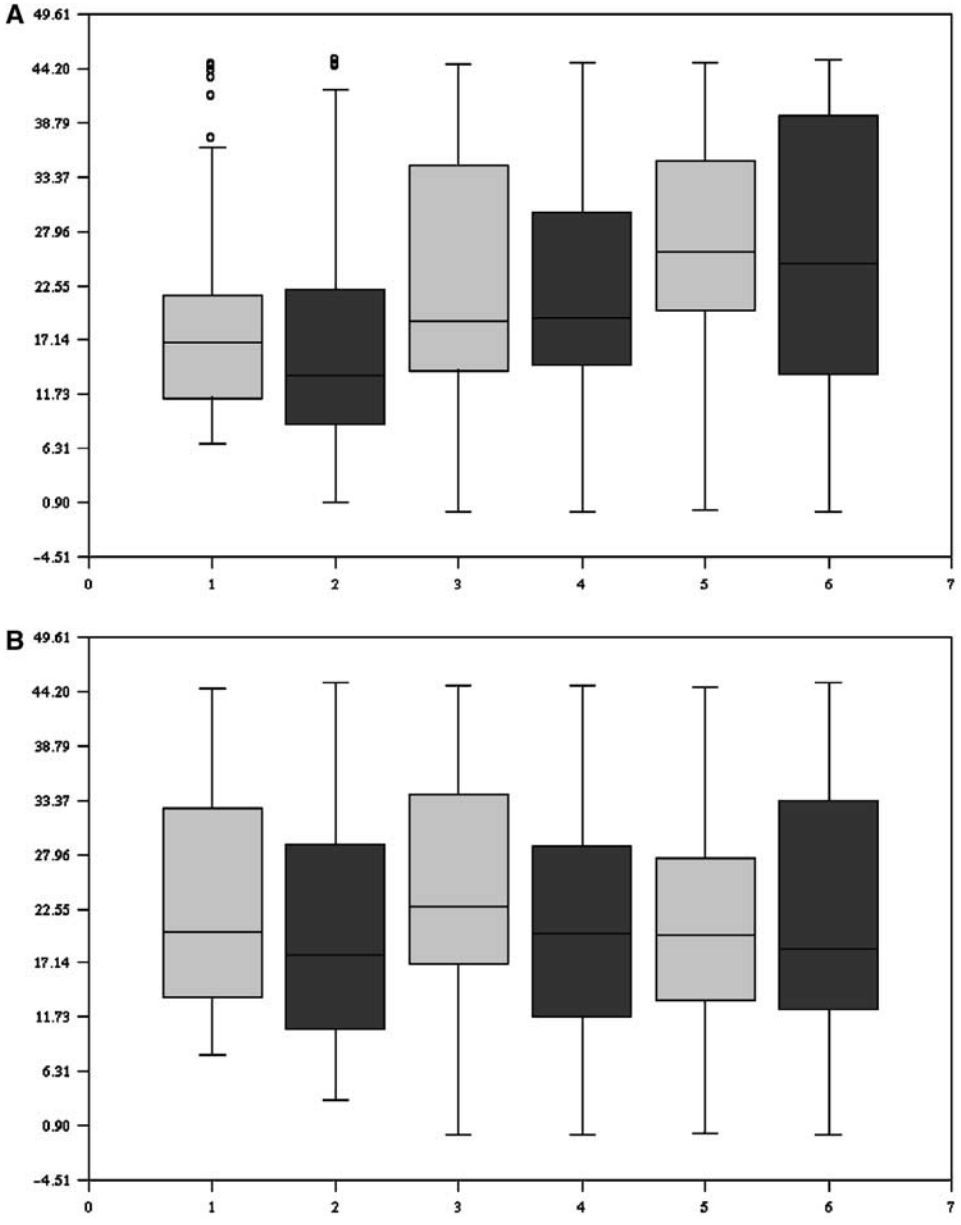

Figure 8 Distribution of answer time for 'mostConnected' (A) split by size, (B) split by density.

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 it (Figure 12A); the answer time does not vary significantly when link density increases (Figure 12B). 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%.

The regression equations related to Figure 13 are:

$$T_{\mathrm{MX}} = 13.3781 - 9.7559 \times d - 0.0714139 \times s + 0.188828 \times d \times s, \tag{12}$$

$$T_{\mathrm{NL}} = 0.088413 \times s. \tag{13}$$

Eq. (13) confirms what we had already observed (Figure 12), that is, answer time using node-link diagrams depends significantly on size only. We could also observe how the answer time using node-links takes off for large values of size and density. As for matrices, an interaction

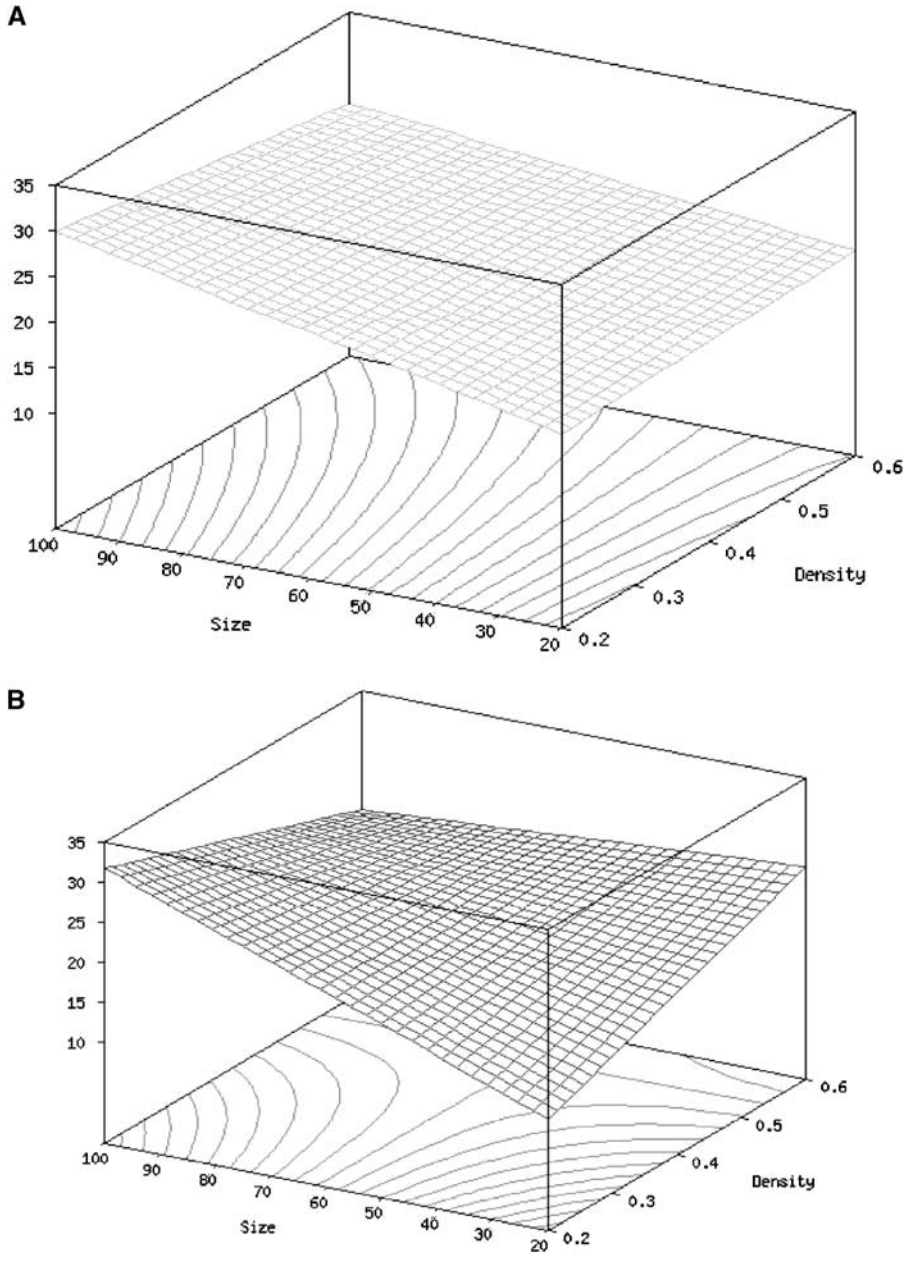

Figure 9 Model of response time with regard to mostConnected for (A) matrices and (B) node-links.

between size and density influences answer time. We also note an improvement in answer time when size or density increase. This may be accounted for by a learning effect relevant for the graphs appearing in phase 1 of the evaluation.

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 14A); the median answer time is marginally affected when link density increases (Figure 14B). 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 (i.e. node-links recorded 84% 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.

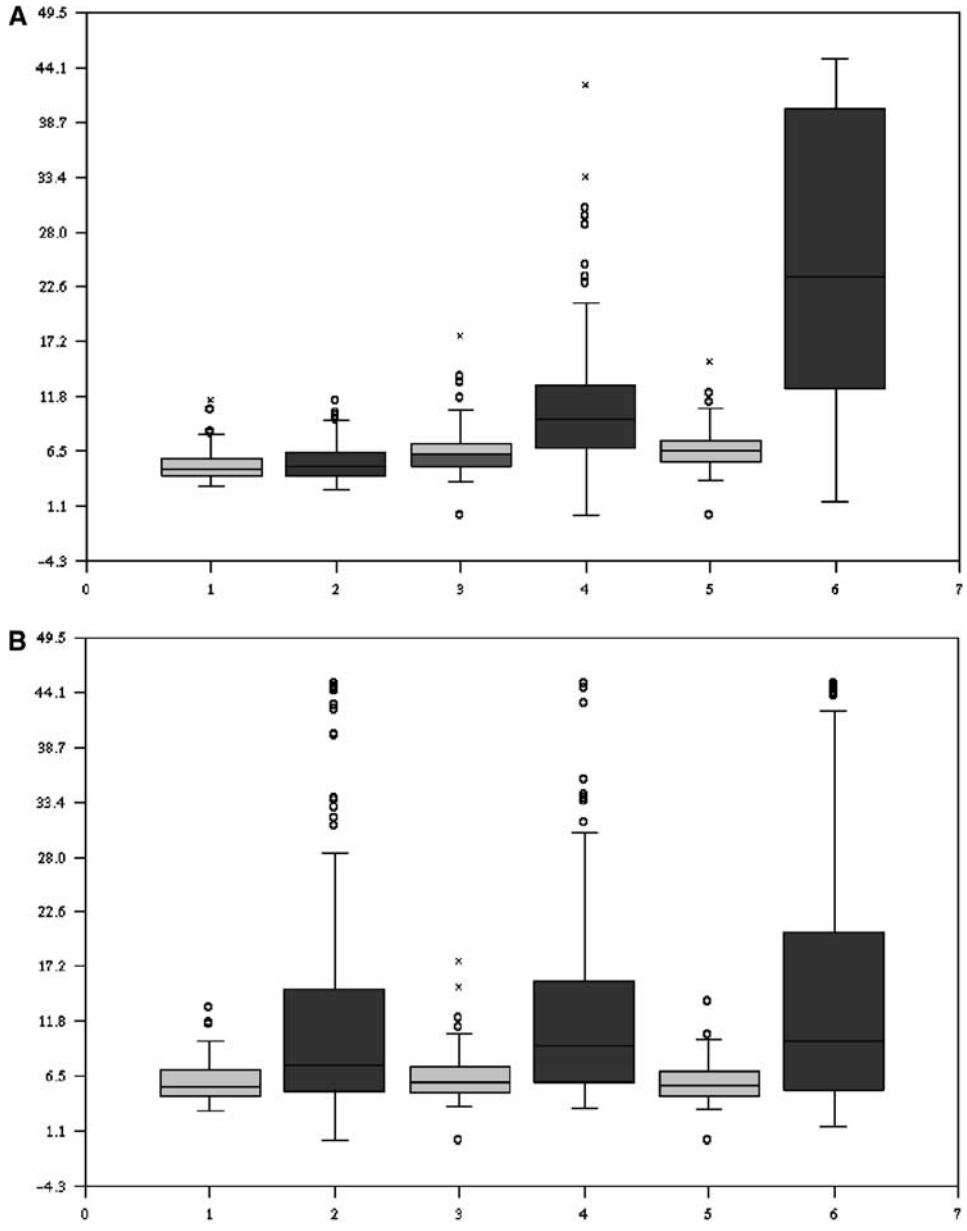

Figure 10 Distribution of answer time for 'findNode' (A) split by size, (B) split by density.

The regression equations plotted in Figure 15 are:

$$T_{\mathrm{MX}} = 15.777, \tag{14}$$

$$T_{\mathrm{NL}} = 0.109657 \times s. \tag{15}$$

Eq. (14) suggests that answer time using matrices is independent of the variables of the problem. At first, this may come as a surprise, but a sensible explanation can be found along the following observations. During the evaluation, the vertices in the questions are automatically highlighted, that is, the related columns in the matrix are highlighted, finding a common neighbor amounts to finding a row intersecting both columns in a pair of black cells, one on each column. Therefore, most users would carry out this task by rolling the mouse over the rows sequentially until they hit one such row or hit the end of the matrix (We may mention once again that when the mouse flies over a row it is highlighted). Very often users would skip the first few rows and then backtrack over them if they fail to find a common neighbor at the first trial. Not only may this schema explain why answer time appears to be independent of size and density, but also why answer time is rather long (around 15 s) compared to other tasks like findNode where answers fall twice as quickly (around 6 s). Using node-link diagrams, answer time depends significantly

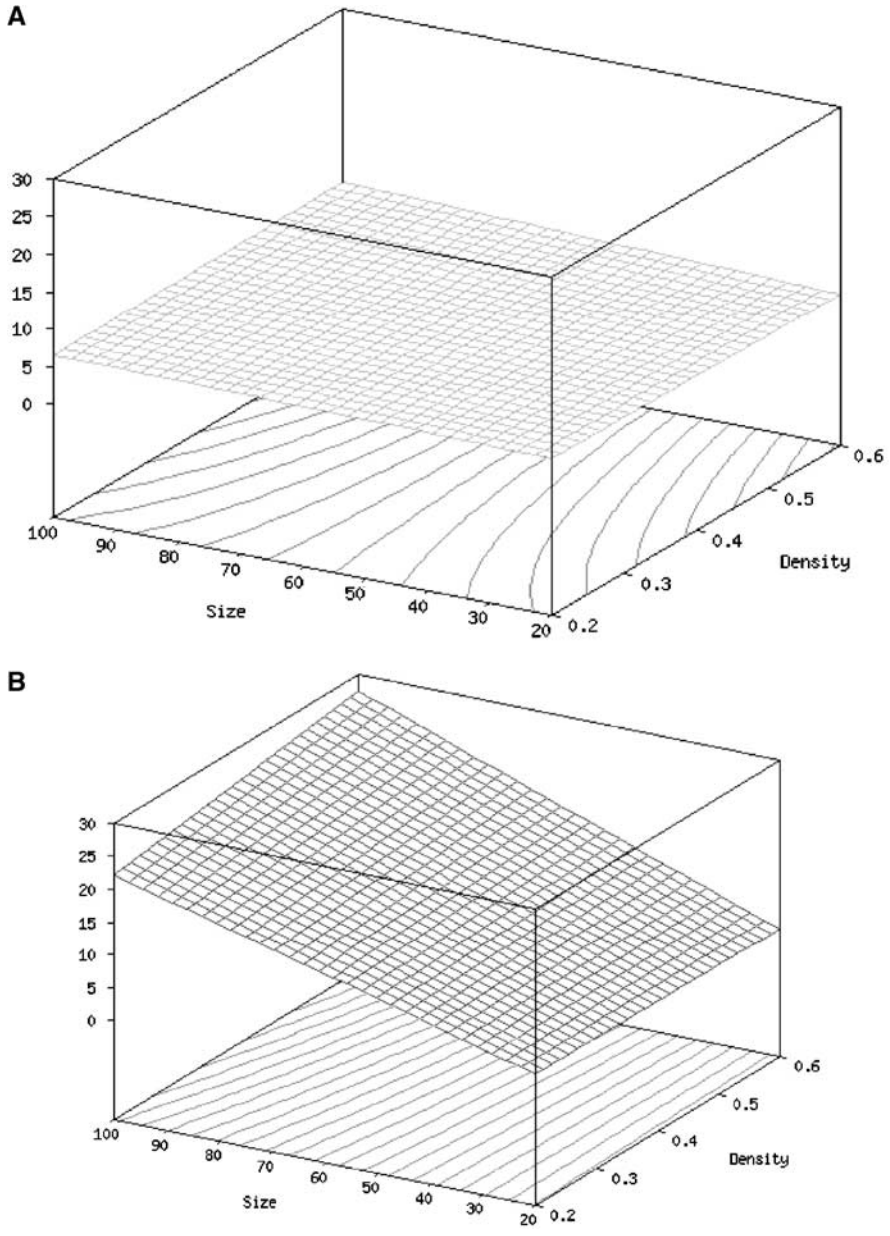

Figure 11 Model of response time with regard to findNode for (A) matrices and (B) node-links.

on graph size, which confirms our previous observations (Figure 14). Node-link diagrams appear to perform better on small sparse graphs, but their performance worsens dramatically with large and dense graphs (see the peak reached on the 3D plot at the high end of size and density values).

Finding a path between two nodes (findPath) Finding a path between two nodes proves to be increasingly difficult using node-link diagrams when the size increases (Figure 16A), whereas the median answer time increases slightly when link density increases (Figure 16B). The matrix-based representation performs very poorly for this task except for very dense graphs where it outperforms the node-link representation (Figure 16B). 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

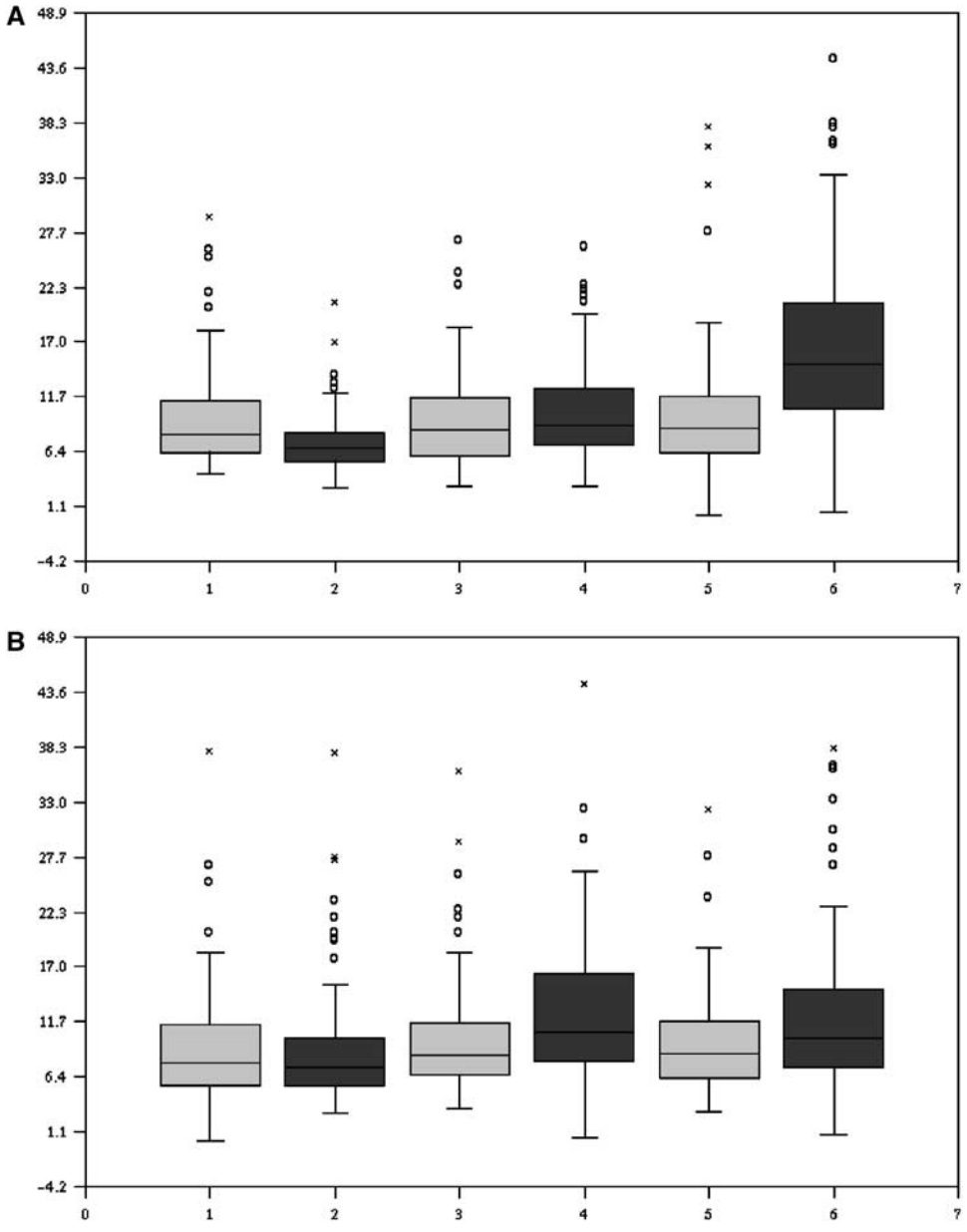

Figure 12 Distribution of answer time for 'findLink' (A) split by size, (B) split by density.

statistically insignificant lead in favor of matrices. Lastly, this task is clearly in favor of node-link diagrams when visualizing sparse graphs.

The regression equations plotted in Figure 17 are

$$T_{MX} = 45.7229 - 53.507 \times d, \tag{16}$$

$$T_{NL} = -6.05372 + 23.2979 \times d + 0.445508 \times s - 0.397442 \times d \times s. \tag{17}$$

The performance of matrices improves significantly when it comes to finding a path as graphs become increasingly dense. This does not come as a surprise since more and more short paths are being available between any pair of nodes. This task would then be quite similar to finding a common neighbor or even finding a direct link between the given endpoints. The answer time expected with node-link diagrams depends on both size and density, while an interaction between size and density may prove helpful. As seen earlier, node-links perform far better than matrices on this task with regard to small sparse graphs. However, matrices outperform them about large and dense graphs.

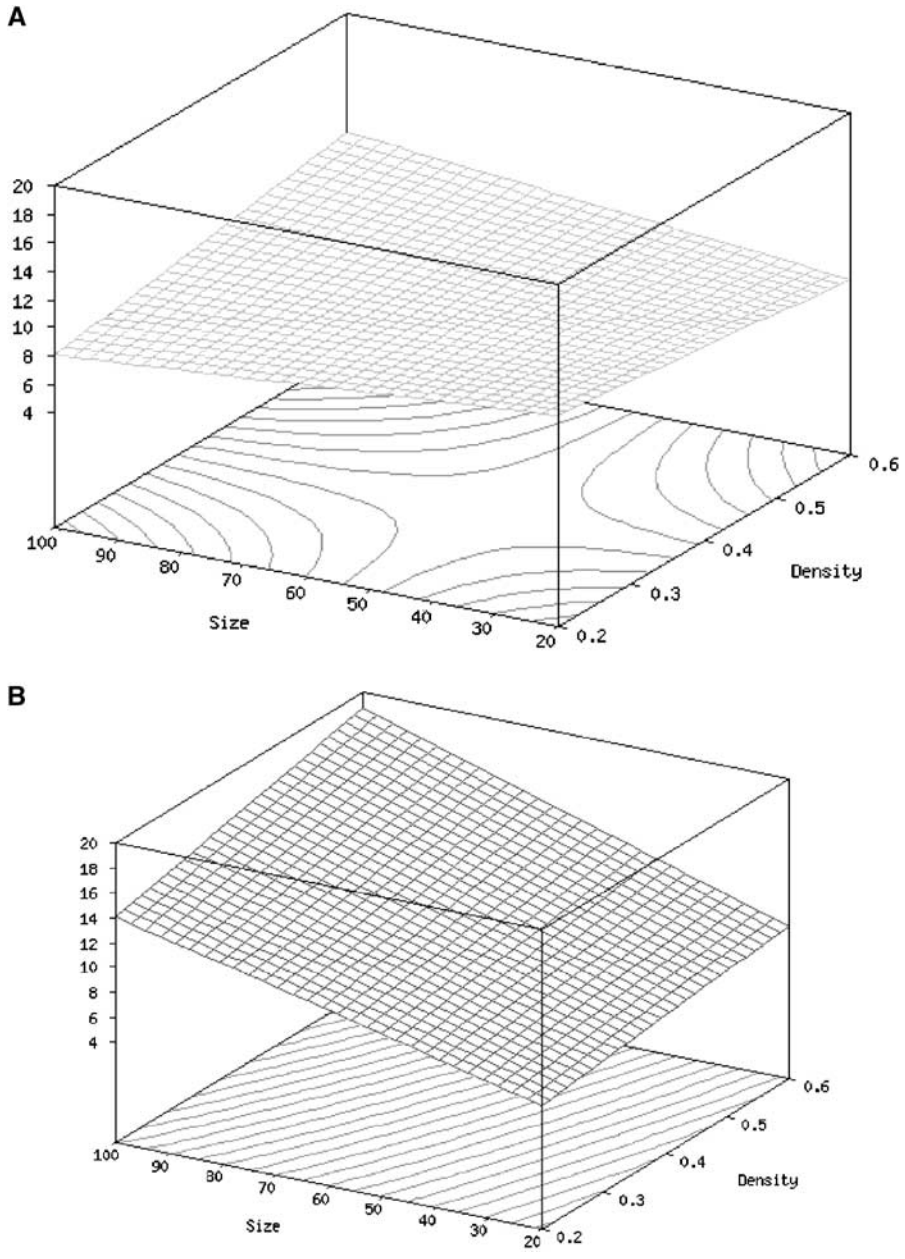

Figure 13 Model of response time with regard to findLink for (A) matrices and (B) node-links.

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 end points is greater than two or three arcs, as shown in Purchase *et al*.[4]

Another hypothesis was related to the significant 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 – 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.

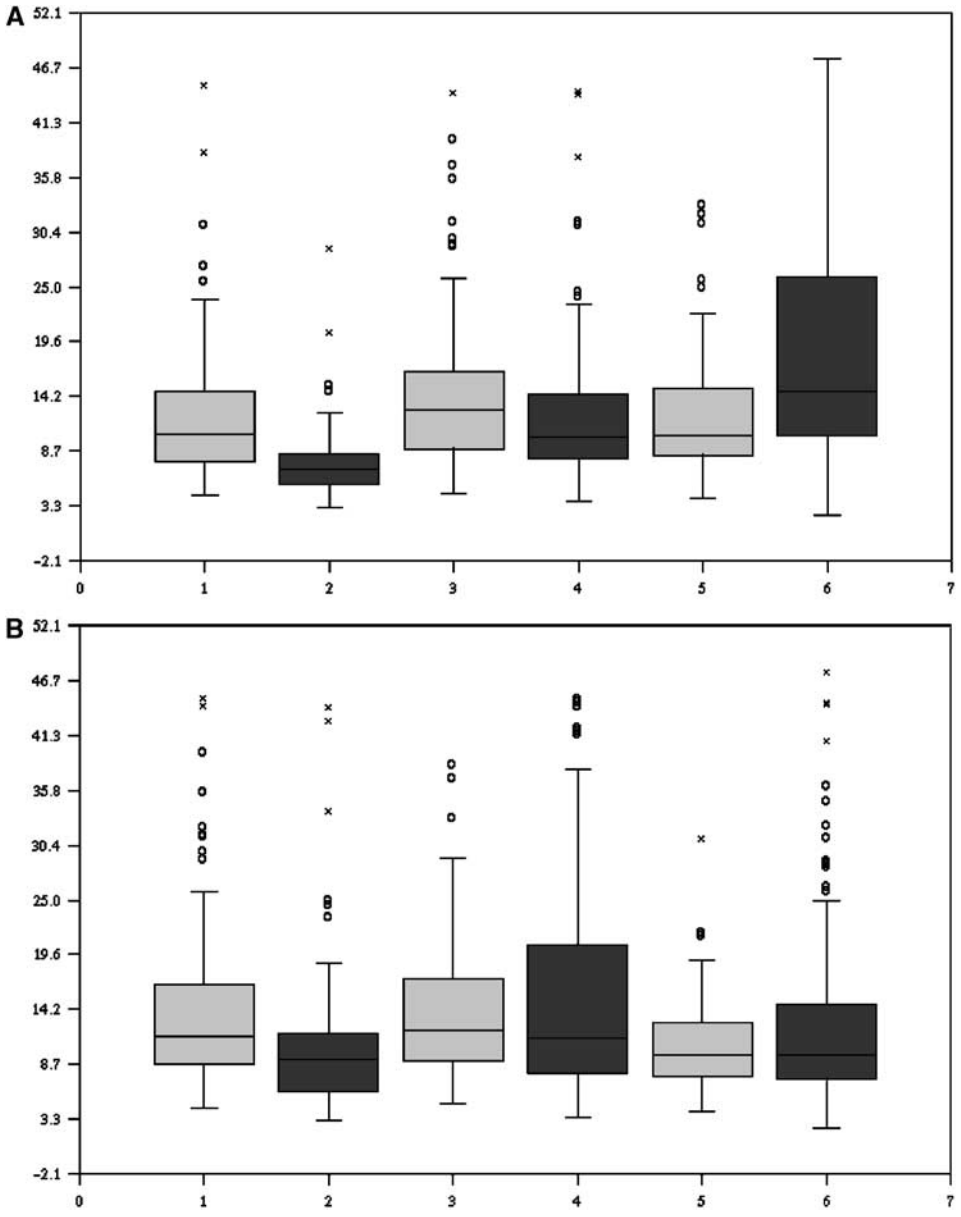

Figure 14 Distribution of answer time for 'findNeighbor' (A) split by size, (B) split by density.

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,[3] 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 visualization of graphs. Since they were given little training through a short demonstration (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. Moreover, a multiple regression analysis has been used to investigate the interaction between size and density of graphs and the impact of such interaction on the readability of their representations. In most cases, no interaction was found for the matrix-based representation, while some interaction was found between size and density with regard to node-link diagrams. A

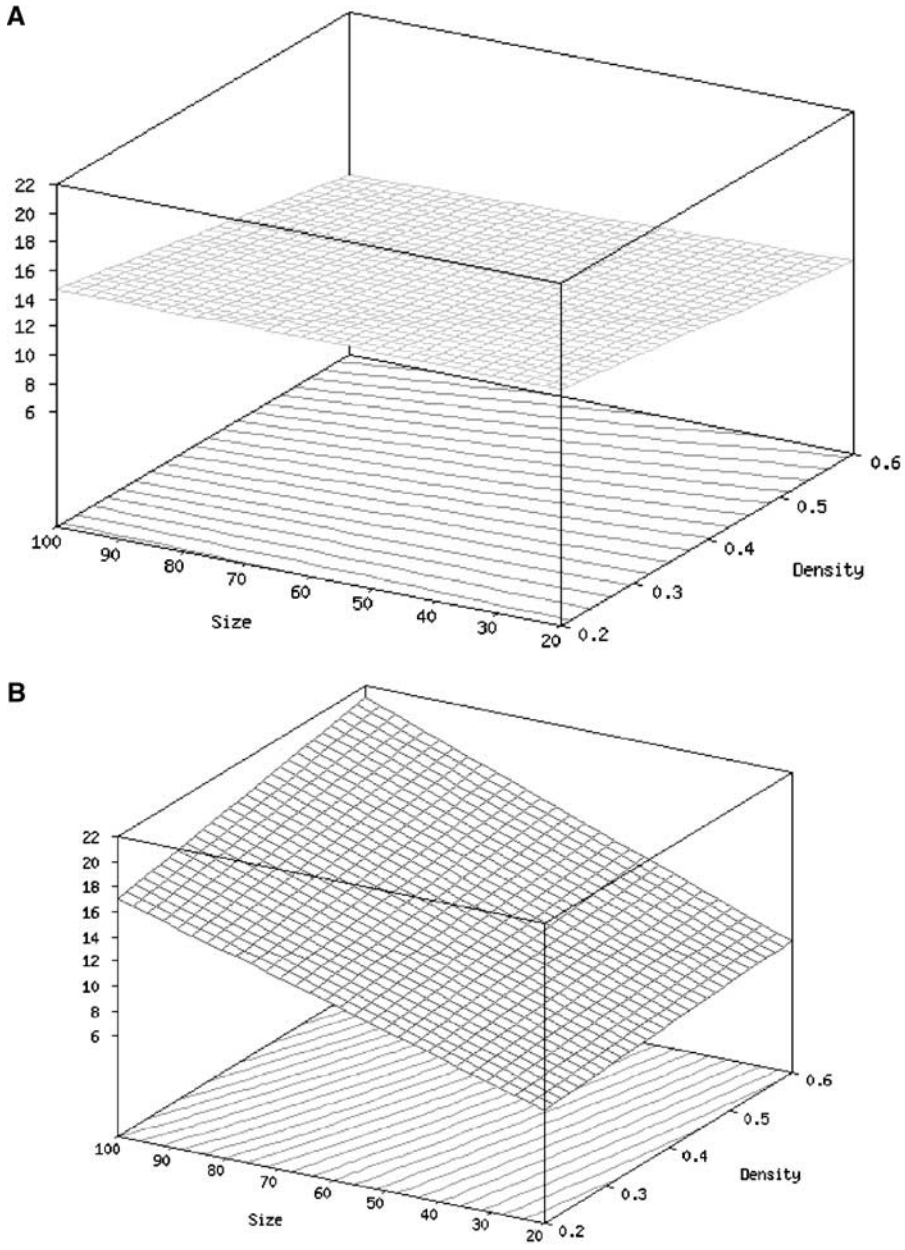

Figure 15 Model of response time with regard to findNeighbor for (A) matrices and (B) node-links.

negative interaction was observed indeed on node-link diagrams with regard to the linkCount, mostConected, and findPath tasks, which means that answer time would improve while the complexity of graphs increase. This can be explained by the fact that node-link diagrams become so cluttered that users would be tempted to give any answer. This is confirmed by the high rate of erroneous answers observed for linkCount and mostConnected. As far as findPath is concerned, many users answered this task correctly. Therefore, there must be another explanation to account for that. For instance, finding a path on a node-link diagram representing a sparse graph proves all the more difficult if the shortest path between the end points is long (which is even more difficult with matrices). In this case, the density of the graph may not be the most relevant 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 renders this task infeasible, while matrices perform very well. Only one task, findLink, related answer time using matrices to both variables of the problem as well as an interaction between them. Otherwise, answer time using matrices would depend on either variable alone, while it occurred twice – for tasks findNode and

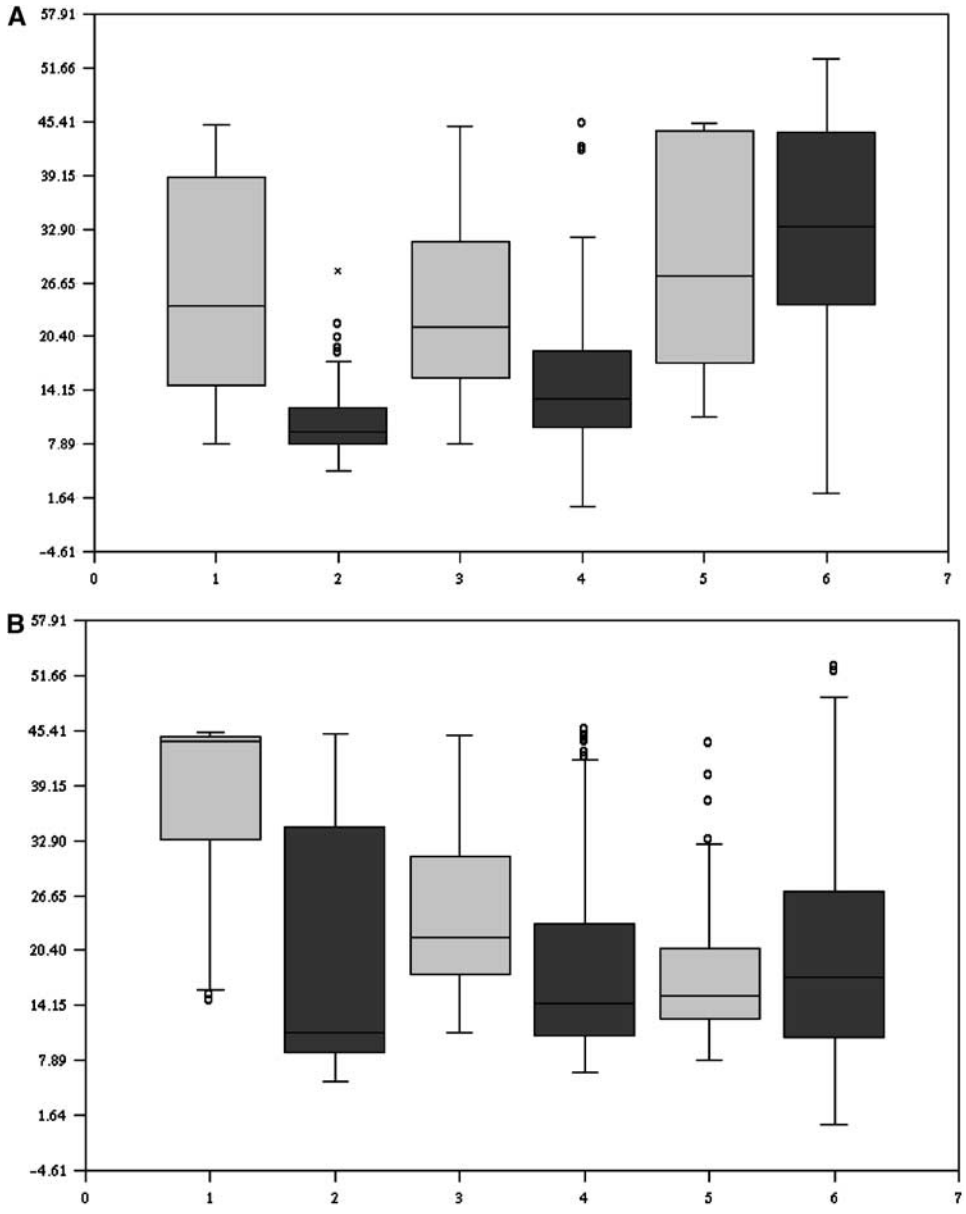

Figure 16 Distribution of answer time for 'findPath' (A) split by size, (B) split by density.

findNeighbor – that answer time was deemed independent of size and density of graphs. This was expected with regard to findNode and came as a surprise with regard to findNeighbor. For the latter, the explanation may be related to the search schema applied by most users (a cursory sequential scan over the rows of the matrix).

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, various recommendations and insights are derived from our study. 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, that is, connecting the cells representing those links.

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. These techniques proved to be complementary: node-link diagrams are well

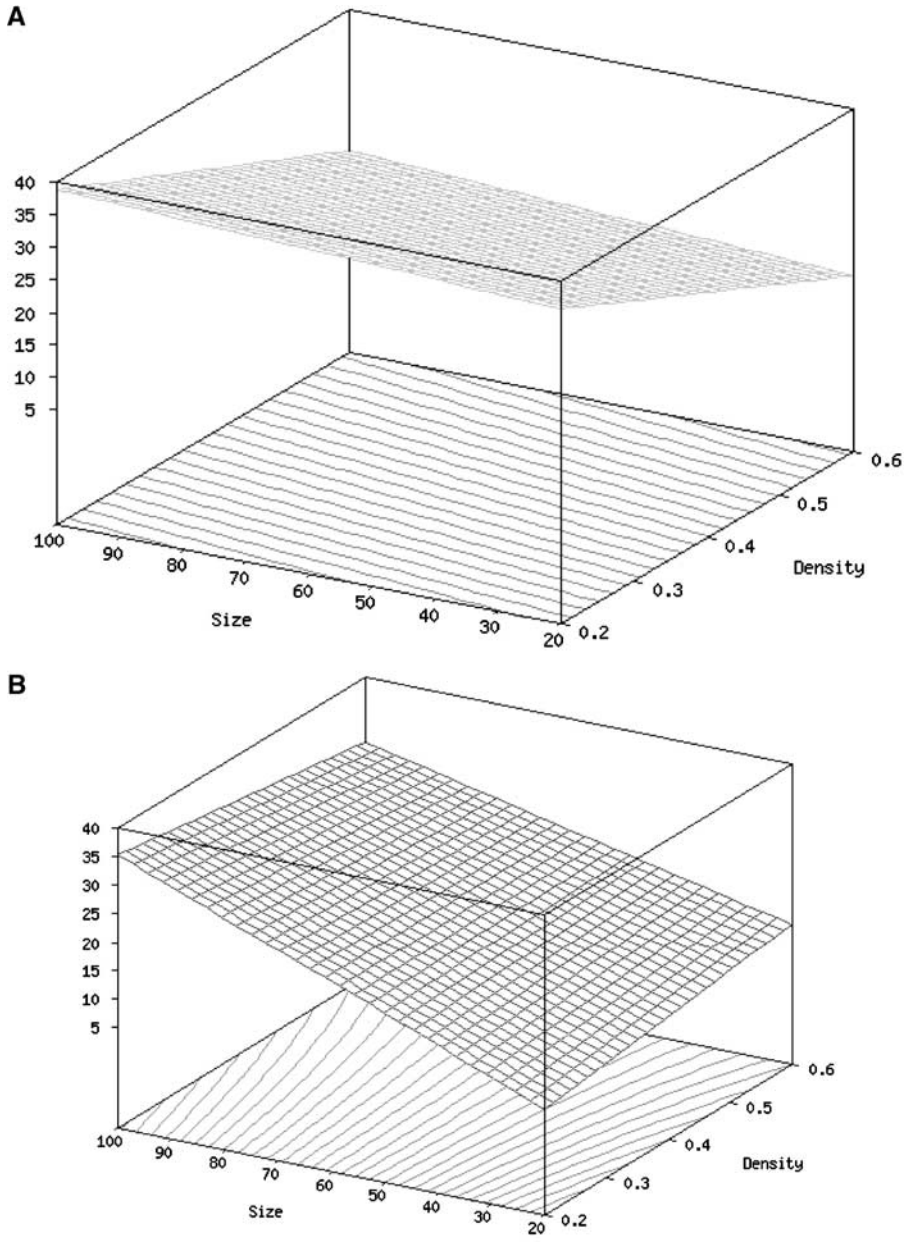

Figure 17 Model of response time with regard to findPath for (A) matrices and (B) node-links.

suited for small graphs, and matrices are suitable for large or dense graphs. Path-related tasks remain difficult on both representations and require an appropriate interaction that helps perform them. We are investigating with interaction techniques to overcome this pitfall.

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 on matrices for the visualization of very large graphs having about tens of thousands vertices.

Acknowledgments

This work has partially been funded by the French RNTL OADYMPPAC project. We thank Pierre Dragicevic, Véronique Libérati and Vanessa Tico for their time and advice. We are grateful to our colleagues at Ecole des Mines de Nantes who volunteered and took part in this evaluation.

References

1 Battista GD, Eades P, Tamassia R, Tollis IG. *Graph Drawing*. Prentice-Hall: Englewood Cliffs, NJ, 1999.
2 Purchase HC, Cohen RF, James MI. An experimental study of the basis for graph drawing algorithms. *The ACM Journal of Experimental Algorithmic* 1997; **2**(4) ACM Press: New York, NY, USA.
3 Purchase HC. The effects of graph layout. in: *Proceedings of the Australasian Conference on Computer Human Interaction* 1998 (Washington, DC, USA), IEEE Computer Society, 80.
4 Purchase HC, Carrington DA, Allder J-A. Empirical evaluation of aesthetics-based graph layout. *Empirical Software Engineering* 2002; **7**: 233–255.
5 Ware C, Purchase HC, Colpoys L, McGill M. Cognitive measurements of graph aesthetics. *Information Visualization* 2002; **1**: 103–110.
6 Cohen RF, Eades P, Lin T, Ruskey F. Volume upper bounds for 3D graph drawing. in: *Proceedings of the 1994 conference of the Centre for Advanced Studies on Collaborative research* 1994 (Toronto, Ontario, Canada), IBM Press.
7 Herman I, Melançon G, Marshall MS. Graph visualization and navigation in information visualization: a survey. *IEEE Transactions on Visualization and Computer Graphics* 2000; **6**: 24–43.
8 Ghoniem M, Fekete J-D, Castagliola P. A comparison of the readability of graphs using node-link and matrix-based representations. *Proceedings of the 10th IEEE Symposium on Information Visualization* (InfoVis'04), 2004 (Austin, TX), IEEE Press: New York, 17–24.
9 Bertin J. *Sémiologie graphique : Les diagrammes – Les réseaux – Les cartes. Editions de l'Ecole des Hautes Etudes en Sciences* 1967 (Paris, France).
10 Becker RA, Eick SG, Wilks AR. Visualizing network data. *IEEE Transaction on Visualizations and Graphics* 1995; **1**: 16–28.
11 Ham FV. Using multilevel call matrices in large software projects. in: *Proceedings of the IEEE Symp. Information Visualization* 2003 (Seattle, WA, USA), IEEE Press: New York, 227–232.
12 Ghoniem M, Jussien N, Fekete J-D. VISEXP: visualizing constraint solver dynamics using explanations. in: *FLAIRS'04: Seventeenth international Florida Artificial Intelligence Research Society conference* 2004 (Miami Beach, FL), AAAI press.
13 AT&T Labs Research Graphviz - open source graph drawing software, 2004http://www.research.att.com/sw/tools/graphviz/.
14 ISPT, Waseda University Random Graph Server.http://www.ispt.waseda.ac.jp/rgs/index.html.
15 Montgomery DC, Runger GC. *Applied Statistics and Probability for Engineers*. Wiley: New York, 1999.