

# Design and Evaluation of MagnetViz—A Graph Visualization Tool

Andre Suslik Spritzer and Carla Maria Dal Sasso Freitas

**Abstract**—MagnetViz was designed for the interactive manipulation of force-directed graph layouts, allowing the user to obtain visualizations based on the graph topology and/or the attributes of its nodes and edges. The user can introduce virtual magnets anywhere in the graph and these can be set to attract nodes and edges that fulfill user-defined criteria. When a magnet is placed, the force-directed nature of the layout forces it to reorganize itself in order to reflect the changes in the balance of forces, consequently changing the visualization into one that is more semantically relevant to the user. This paper describes MagnetViz's concepts, illustrating them with examples and a case study based on a usage scenario. We also describe how the MagnetViz has evolved since its original version and present the evaluation of its latest version. This evaluation consists of two user studies aiming at assessing generated layout quality and how well the concepts can be apprehended and employed, and a task taxonomy assessment focusing on establishing which graph visualization tasks the technique is able to handle.

**Index Terms**—Graph visualization, force-directed layout, evaluation, social networks visualization.

## 1 INTRODUCTION

GRAPHS are widely used to represent information in many fields. Although there are other approaches, the most common visual representation of a graph is the node-link diagram, with many layout algorithms having been created to draw them, each favoring different aesthetic properties [1]. Works in graph visualization very often entail one of these algorithms associated with a navigation and interaction scheme that allows users to explore and analyze the graph [2].

One of the classic families of graph drawing algorithms is the one of the force-directed layouts. These algorithms treat the graph as a physical system, assigning forces to nodes and edges and minimizing the energy until a stable layout is reached. The resulting layout is clean and organic-looking, with few edge crossings and more-or-less uniform edge lengths. Being essentially physics simulations, they are very flexible in the application of constraints, since anything that can be expressed by forces can be integrated into the algorithm. All of these characteristics and their relative ease of implementation have made them very popular despite the fact that they can be computationally expensive.

Force-directed layouts, though, as most other layout algorithms, usually take into consideration only the topology of the graph, ignoring its *metainformation*—that is, the attributes of the nodes and edges. While a graph's topology

is undoubtedly its main feature, its metainformation can play a crucial role in many applications such as social network analysis [3] and computer networks [4]. Still, with a few exceptions, even graph visualization in general has given metainformation little emphasis, focusing more on displaying, navigating, and analyzing the topology of the graph.

While generic graph visualization techniques and layout algorithms are mostly effective at generating high quality visualizations from an aesthetic and interaction standpoint, the fact that a graph's metainformation is usually not taken into account makes them insensitive to the context in which they are used. This is an advantage in the sense that the techniques can be applied in a wide gamut of fields, but a disadvantage in the sense that users might not get the visualization most adequate for their needs. Users, then, might feel the necessity to manipulate a visualization's automatically generated layout into one better suited for their specific contexts.

Many graph visualization techniques allow for some manipulation of the layout by letting users group nodes, move them around, or fix them to certain positions (see, for example, [5]). Other techniques do not allow for direct layout manipulation, but change a graph's original layout by creating recursive clusters of nodes based on their metainformation. These clusters can be precomputed and then explored or, in steerable techniques, created on the fly as a user navigates a graph. These approaches are however limited in the sense of providing users with tools to affect a graph's layout and visualization on a large scale.

With these issues in mind, in a previous work [6] we introduced a graph visualization technique called MagnetViz. MagnetViz was designed with the goal of allowing users to interactively manipulate a force-directed graph layout into one more suitable to their needs by using tools that seize both on the graph's topology and application-specific semantics. The resulting layout can lead to a more expressive visualization that may provide new insight into

• A.S. Spritzer is with the Universidade Federal do Rio Grande do Sul (UFRGS), Rua Artur Rocha 812/901, 90450-170 Porto Alegre, Rio Grande do Sul, Brazil. E-mail: spritzer@inf.ufrgs.br.

• C.M.D.S. Freitas is with the Universidade Federal do Rio Grande do Sul (UFRGS), UFRGS - Instituto de Informática, Av. Bento Gonçalves 9500, Caixa Postal 15.064, 91.501-970 Porto Alegre, Rio Grande do Sul, Brazil. E-mail: carla@inf.ufrgs.br.

Manuscript received 18 Mar. 2010; revised 20 Apr. 2011; accepted 5 June 2011; published online 16 June 2011.

Recommended for acceptance by S. Carpendale.

For information on obtaining reprints of this article, please send e-mail to: [tcvg@computer.org](mailto:tcvg@computer.org), and reference IEEECS Log Number TVCG-2010-03-0071. Digital Object Identifier no. 10.1109/TVCG.2011.106.

the data set and serve as a starting point for further exploration of the graph.

In that previous work [6], we have described MagnetViz and its first prototype and presented a brief assessment of the technique against a task taxonomy [7] to verify which basic common graph visualization tasks it could help users perform. MagnetViz continued to evolve, undergoing several evaluation and refinement stages.

This paper reports the final version of MagnetViz, detailing its concepts, implementation, and evolution, and describing the evaluation process to which it was submitted and the results obtained.

## 2 RELATED WORK

### 2.1 Attribute-Based Graph Visualization

In most graph visualization techniques, attributes are handled in a complementary fashion. They can serve as labels, be encoded as different shapes, colors, and sizes for nodes and edges, or be used for filtering and selection. The most influence attributes usually have on a layout is when they are used to create metanodes in order to decrease the visualization's cognitive overload. These metanodes often contain other metanodes, becoming a graph hierarchy that can be explored by users by means of interaction and navigation techniques [8].

While most graph visualization techniques are still focused on topology, recent years have witnessed a growth of attribute-based techniques. These techniques build visualizations based on the attributes with the topology serving only in a limited way (if used at all), borrowing heavily from the field of multidimensional data visualization [9], [10], [11], [12].

A recurring theme in attribute-based visualization is the use of scatterplots, like the ones provided in techniques such as Wattenberg's PivotGraph [9], Aris and Shneiderman's semantic substrates [10], [11] and Bezerianos et al.'s GraphDice [12]. In essence, the basic approach consists in displaying nodes as points in a scatterplot that has as its axes two user-chosen attributes, with edges shown as lines connecting the nodes. Despite the same basis, these scatterplot-based techniques have important differences.

PivotGraph [9] displays one scatterplot at a time, with nodes that have the same value for the same categorical attribute being aggregated in metanodes that are represented as points with size proportional to the number of nodes they contain. The semantic substrates technique [10], [11], on the other hand, uses no aggregation but displays several scatterplots simultaneously. Nodes are placed into regions, each containing a scatterplot, and users are provided with controls to define the visibility of edges. Yet another approach is taken by GraphDice [12], which has two panes, one that displays the main plot and another for an overview matrix of all possible scatterplots (rows and columns containing all attributes and a thumbnail of each scatterplot drawn at the intersections). The main plot changes by navigating in the scatterplot matrix, with animation providing smooth transitions from one layout to the next. None of these techniques actually handles edge attributes.

While scatterplots are so far the predominant approach to attribute-based graph visualization, they are not the only

option. Pretorius and van Wijk [13] proposed a technique for the visualization of transition graphs based on hierarchical attribute-based clustering. Clusters are positioned linearly, with their hierarchy shown as a tree-like node-link diagram and edges shown with a superimposed arc diagram. Later on, the same authors proposed another system for multivariate graph visualization based on recursive attribute-based clustering [14]. The cluster hierarchy is displayed on the left and right sides of the screen, respectively, representing the source and target nodes, while the edge labels are displayed in the center, with lines linking source and target nodes to their respective edge labels.

The aforementioned approaches have three major issues:

- they focus too much on the attributes in detriment of the topology. This is illustrated by the scatterplot techniques: by using purely scatterplots to lay out a node-link diagram, the structure of the graph is not necessarily made clear, since the positioning is based only on node attributes.
- they pay too little attention to edges and their attributes. For example, all the previously mentioned techniques, with the exception of GraphDice, only handle nodes.
- they are either simple but limited or powerful but too complex. In terms of power and cognitive load, visualizations such as PivotGraph are simple to use and comprehend but limited in what they can actually show, while others are powerful but a lot more complex, requiring a more specialized user (e.g., semantic substrates [10], [11], Pretorius and van Wijk's techniques [13], [14]).

Aside from these "purely" attribute-based techniques, there are some others that take a middle-ground approach, trying to balance topology and attributes. One such technique is GrouseFlocks [15], which builds a topology-based layout on top of an attribute-based cluster hierarchy refined by the users. Another "middle-ground" technique is MagnetViz itself, which provides users with tools to manipulate a layout according to their own criteria, either in terms of the graph's topology or meta-information.

### 2.2 Interactive Manipulation of Graph Layouts

Node-link layouts automatically generated based on the graph topology might not be sufficiently adequate for users of certain applications. These users may want to manipulate these layouts into ones that are more meaningful for their specific necessities. Traditionally, limited layout modification is allowed, users being able to move nodes around and fix them to specific positions. However, that is often not enough to address users' needs.

Dynamic, interactive layout modification can be found in the hybrid techniques that combine node-link diagrams with other representations. One of these is Henry et al.'s NodeTrix [5], in which users can choose to have groups of nodes visually represented by either matrices or node-link diagrams, with both representations shown in a single, integrated view. Another is Zhao et al.'s Elastic Hierarchies [16], which combine treemaps and node-link diagrams in the same fashion. These techniques manage to combine the

intuitiveness of the node-link diagram representation with the power of representations that are integrated with it.

Instead of integrating different representations into the same visualization, one can also integrate different layout algorithms. McGuffin and Jurisica [17] proposed an approach to allow users to modify a node-link diagram by letting them select a subgraph in a node-link diagram and apply a layout algorithm that will act only on its nodes, with the remainder of the graph maintaining the original layout. By focusing on the layout of subgraphs, this technique can be said to take a local approach to layout manipulation. MagnetViz, on the other hand, while also allowing for the selection of subgraphs, focuses on the manipulation of the layout as a whole, taking a global approach. Even when only a subgraph is actively manipulated by a user, the entirety of the graph might change to reflect the repositioning of this subgraph's nodes.

Focused mainly on the problem of scaling (i.e., laying out large graphs), techniques have been proposed that take a precomputed graph layout and use it to build a cluster hierarchy that will be interactively explored by navigating through recursive metanodes. Steerable techniques are a variation of this approach, building the visualization as the user explores the graph. Several examples of these techniques are given in Archambault et al.'s paper on the readability of path-preserving clusterings of graphs [18]. Particularly, interesting among these techniques is Archambault et al.'s Grouse Flocks [15], which take an approach similar to MagnetViz by allowing users to dynamically and interactively edit the attribute-based graph hierarchy in order to make it suit their specific needs.

### 2.3 Metaphor

Metaphors are often used in visualization techniques to convey information and express functionality with a low cognitive effort. In this work, we use a magnet metaphor that was also used in Dust & Magnet [19]. However, aside not being designed for graphs, it is not physically based. Moreover, magnets are predefined for each numerical attribute, with data particles moving more or less toward them based on attributes values. In MagnetViz, magnets act physically in the layout as we describe in next section.

## 3 MAGNETVIZ

This section presents MagnetViz, describing its concepts, how they were implemented, as well as the evolution of our visualization tool.

### 3.1 The Magnet

Being the key concept in MagnetViz, magnets are special objects that can be added to the scene to attract nodes of a graph that fulfill certain user-defined **criteria**. A magnet works by exerting onto each of these nodes an attraction force that will progressively move them toward it, thereby building a cluster of semantically related nodes around it. When these nodes move, the force-directed layout algorithm ensures that all the other nodes that are connected to them by edges will also move, reorganizing the whole layout of the graph in the process.

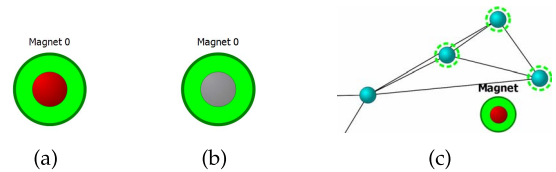


Fig. 1. Visual representation of magnets and their attracted nodes: (a) an active magnet; (b) an inactive magnet; and (c) three nodes being attracted by a magnet.

When creating a magnet, users should assign to it at least one attraction criterion and define the magnitude of the attraction force. If a magnet has multiple criteria, any node that fulfills any of the criteria will be attracted. If users want to make sure that attracted nodes fulfill a specific criterion, though, they can set it as a **requirement**. In order to be attracted, a node needs to satisfy all requirements and at least one of the remaining criteria (if there are any). Simple criteria work, then, as the logical operator “OR,” while criteria marked as requirements work as the operator “AND.” The “NOT” operator is also supported by marking a criterion as a complement, which will attract all nodes that *do not* fulfill it. Criteria are managed in the magnet's properties panel, where users can also define whether the magnet is active or not (that is, whether it exerts attraction forces on the nodes that fulfill its criteria) and set properties regarding the magnet's visual representation and hierarchy (see Section 3.4).

Magnets are visually represented as shown in Fig. 1. The magnitude of a magnet's attraction force is visually encoded as the magnet's radius, being always proportional to it and altered by dragging the mouse with the shift key and left mouse button pressed. The magnet's inner circle's fill color is red when the magnet is active (Fig. 1a) and gray when it is inactive (Fig. 1b). The outer color is the “magnet color,” which is also used in a dashed ring that appears around attracted nodes (Fig. 1c). When a node is attracted by more than one magnet, rings corresponding to each of the magnets are displayed around it (see Section 3.3).

MagnetViz's flexibility in layout manipulation comes in great part from the different types of attraction criteria that can be employed. We list below the criteria implemented in the latest version. While just a subset of what MagnetViz conceptually supports, they were chosen for the considerable amount and variety of operations they allow for, being good tools to illustrate the technique.

#### Topology-based

- **Degree:** Attracts nodes based on degree (i.e., edge amount).
- **Adjacent nodes:** Attracts nodes based on its number of adjacent nodes.
- **Edge amount:** Attracts nodes based on the amount of edges they have to a given target node.
- **Path:** Attracts nodes based on the topological distance of their paths to a target node.
- **Connected component:** Attracts connected subgraphs (maximally connected or not) based on the number of nodes they should have or on a specific node they should contain.

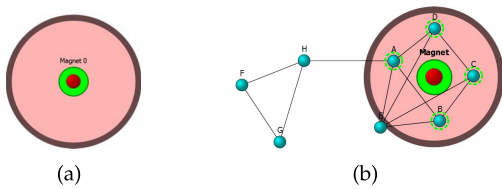


Fig. 2. Boundary shape: (a) a magnet with a boundary shape; (b) a magnet with a boundary shape attracting all of node E's neighbors.

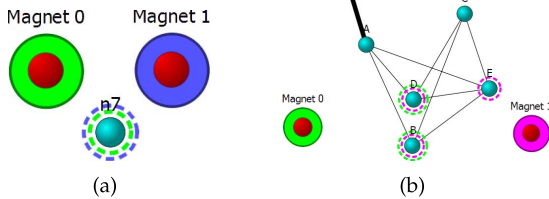


Fig. 3. Magnet intersection: (a) a node and its two magnets and (b) two magnets with two nodes in common.

**Attribute-based**

- **Node/Edge has attribute:** Attracts nodes/edges that possess a chosen attribute, with any value.
- **Node/Edge has attribute with numerical value:** Attracts nodes/edges based on their value for a chosen numerical attribute.
- **Node/Edge has attribute with substring:** Attracts nodes/edges that for the chosen textual attribute have a value containing a user-provided substring.

**Magnet-based**

- **Magnet:** Attracts nodes based on what other magnets attract or not, being particularly useful for set operations such as union and intersection.
- **Node has connection to magnet:** Attracts nodes that have a connection (edge) to other nodes that are attracted by a target magnet.

**3.2 The Boundary Shape**

Depending on their needs, users may want to isolate the nodes attracted by a magnet into a certain region of the workspace. MagnetViz allows them to do that by means of **boundary shapes**. A boundary shape is a geometric shape placed around a magnet that delimits a region where attracted nodes should be. All of these nodes are forced to remain inside the boundary shape, while nonattracted nodes are kept outside. By using a boundary shape, one can better understand the attracted nodes' relations to each other and also how they, as a subgraph, relate to the rest of the graph.

In the MagnetViz tool, boundary shapes are implemented as circles with user-chosen radius and fill color (Fig. 2). In the magnet's properties, users can define whether forces caused by nonattracted nodes will affect those inside the boundary shape or not.

**3.3 Magnet Intersection**

Nodes can be attracted by more than one magnet. We call this a **magnet intersection**. When this happens, dashed rings in the colors of each attracting magnet are displayed around each intersecting node (Fig. 3).

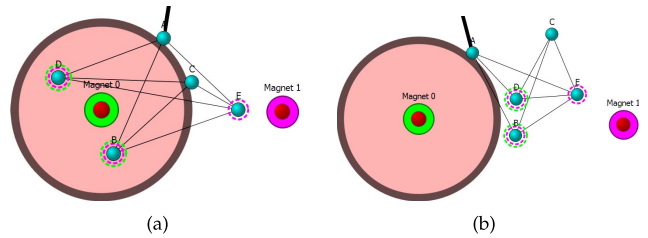


Fig. 4. Behavior of intersecting nodes: (a) bound to the green magnet and (b) left free.



Fig. 5. Magnet hierarchy: a magnet and its child.

When there is a magnet intersection, users must choose the behavior the intersecting nodes will assume. These nodes can be bound to a specific magnet or left free. In the first case, if the binding magnet has a boundary shape, the nodes will act as if they were attracted only by the chosen magnet, ignoring the attraction forces exerted on them by the others (Fig. 4a). In the second case, they will be affected by the forces of all magnets that attract them, but will ignore the restraints imposed by the boundary shapes (Fig. 4b).

In the prototype, the behavior of intersecting nodes is defined in the magnet properties panel. Magnets that intersect with the selected magnet are listed alongside checkboxes that indicate whether the commonly attracted nodes should be bound to the current magnet or not. The default behavior is to leave nodes unbound.

**3.4 Magnet Hierarchy**

When building a layout using MagnetViz, users may want to create a magnet that acts only on a subset of nodes that is already attracted by another magnet. This is accomplished by the creation of **child magnets**, which act only on the nodes attracted by the parent. The connection between a parent and a child magnet is shown as a dotted line with an arrow in the child's end linking it to the parent (Fig. 5). The intersection between parent and child is handled as all others and child magnets can have children themselves. A child magnet is created in the properties panel of its parent magnet. When a parent is deleted, all of its children are as well.

**3.5 MagnetViz Visualization Tool**

In order to illustrate MagnetViz's features and concepts and conduct an evaluation cycle, a graph visualization tool was developed. A standard node-link representation is used, with multiple edges between two given nodes being represented by single edges with variable edge thickness. Aside from MagnetViz's features, standard interaction and exploration techniques (selection, zoom, pan, etc.) are provided, as well as a simple system for color coding and labeling of nodes.

For the graph's layout algorithm an approach similar to Tunkelang's [20] was used. Tunkelang's technique is an adaptation of the widely known Fruchterman-Reingold

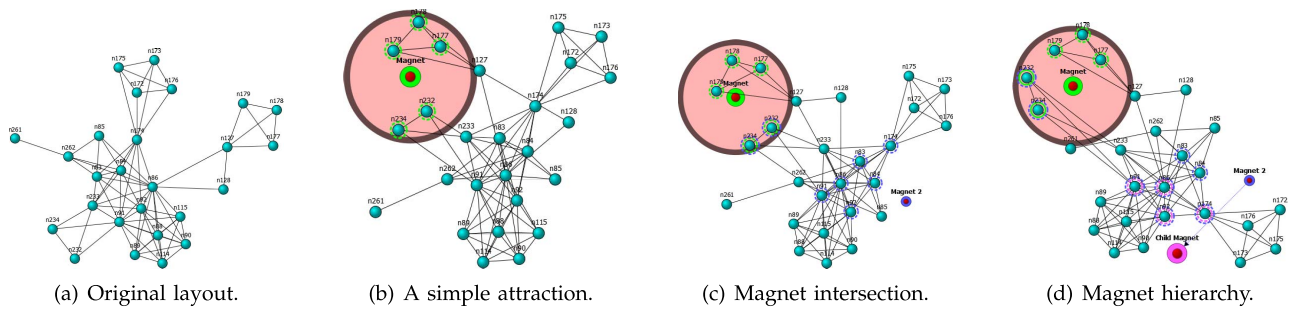


Fig. 6. Using the MagnetViz tools on a small graph.

algorithm [21], producing the same visual results while reducing its complexity from  $O(n^3)$  to  $O(n^2)$ . This method was chosen as a basis for the prototype because it generates aesthetically pleasing layouts and is easy to implement and integrate with MagnetViz. In our version, we added to Tunkelang’s method a small, user-definable gravitational force that pulls nodes slightly toward the center of the workspace and used Verlet’s integration method [22] to keep the physics engine running all the time in order to dynamically respond to the changes in forces that might occur while users interact. An extra step was also added to deal with the position constraints imposed by the boundary shapes.

### 3.6 The Evolution of MagnetViz

The first implementation of MagnetViz [6] and its successive improvements were submitted to complete evaluation cycles. Every iteration was assessed using Lee et al.’s task taxonomy [7] and went through heuristic evaluations and tests with expert and nonexpert users. Whenever user feedback resulted in changes, these were then evaluated, resulting in further refinement and enrichment of the technique. The current MagnetViz version was a result of this process and, as all other iterations, also went through an evaluation phase, this time with an additional test to evaluate the quality of the generated layouts (Section 5.1).

The current visual representation of magnets and attracted nodes is a marked improvement over the earlier versions of MagnetViz [6]. In previous versions, magnet radius and attraction strength had no relation and one fixed icon was used for all magnets, independent of the color chosen by the user. In addition, nonintersecting nodes were filled with the color of their attracting magnet (making color coding impossible), while intersecting nodes used to assume an alternate visual representation that did not indicate the attracting magnets, being linked to those by colored lines that cluttered the visualization. Earlier implementations had also no visual indication of magnet hierarchy, with no visual link between a parent magnet and its children.

Changes were also made to the visual representation of the layout itself. In earlier versions, all nodes that were not attracted by a magnet had the same color and multigraphs were not supported.

The interactive aspect of MagnetViz also evolved. Previously, attraction strength was set numerically, while now it is defined by directly manipulating the radius of the magnet. Criteria and requirements were displayed in two separate

lists and there was no equivalent of a “NOT” operator. Magnet intersections were at first handled by pop-up messages, which were eventually replaced by an “intersection manager” panel that listed all intersections and allowed users to choose a magnet to which nodes could be bound or whether they should remain free. As for boundary shapes, there was no option of ignoring forces caused by outside nodes, with nodes inside it being always affected by all forces. Magnet hierarchies were built by creating a magnet and setting it as another’s child by picking the parent from a combo box in the properties panel.

### 3.7 Examples

In this section, we show how a small graph (Fig. 6a) can be manipulated with magnets. In Fig. 6b, a magnet has been inserted to attract all nodes with degree 3. This magnet’s boundary shape is visible and configured as to allow inner nodes to be affected by outside forces (i.e., if a node outside the boundary shape shares an edge to a node that is inside, they will attract each other). In Fig. 6c, a second magnet has been added to attract nodes that have a connection to node  $n233$ . Since two of the nodes attracted by the first magnet also fulfill this criterion, an intersection happens. This intersection is dealt with by binding the two magnets’ common nodes to the first’s boundary shape. As a result of this, intersecting nodes are kept within the boundary shape. Finally, in Fig. 6d, we give Magnet 2 a child magnet that attracts nodes with degree higher or equal to 10. Notice that this magnet attracts only the nodes that are attracted by its parent, Magnet 2.

An example of MagnetViz in use with a larger graph is given in Fig. 7. Fig. 7a shows the original layout, while in Fig. 7b several magnets are used to separate the connected components of the graph according to how many nodes they have.

## 4 CASE STUDY

In order to illustrate MagnetViz and what can be achieved with it, in this section we present a case study on the visualization of the coauthorship network of the 2009 publications of the UFRGS Informatics Institute students and faculty and their external collaborators. The graph, shown in Fig. 8, has 438 nodes and 1,989 edges, with nodes representing the authors and edges being their common publications. Color coding was used, with pink nodes being faculty members, yellow nodes being students, and blue nodes being external collaborators. For

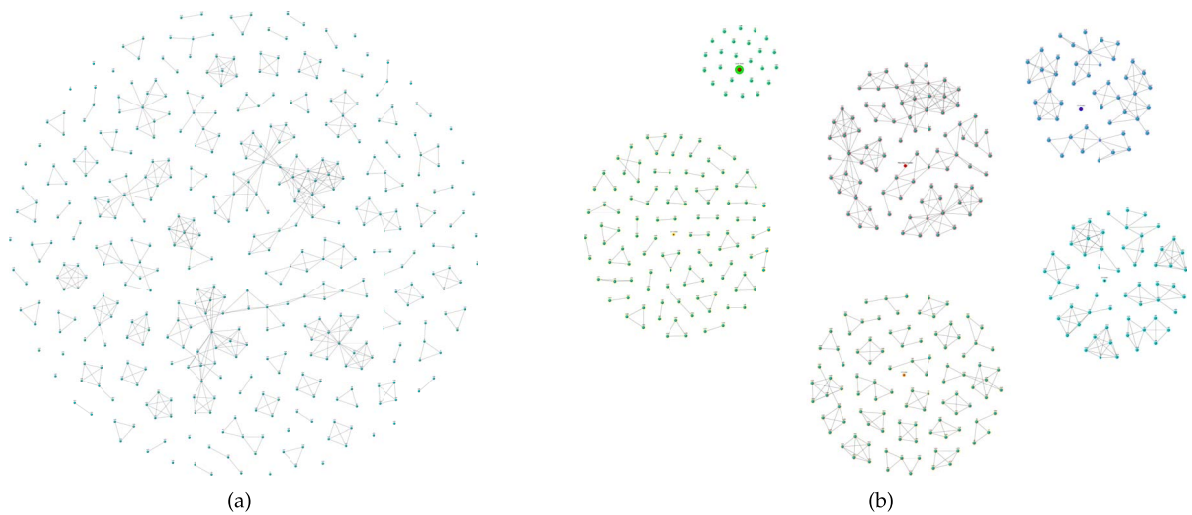


Fig. 7. Using MagnetViz on a larger graph: (a) original layout and (b) magnets are used to separate nodes according to size of connected components.

privacy protection, nodes were labeled with randomly assigned identification numbers.

The case study takes the form of a usage scenario, where one aims at improving the visibility of the collaboration among researchers of the Computer Graphics (CG) group at UFRGS. The tasks performed were based on Lee et al.’s task taxonomy on graph visualization [7].

The graph’s exploration begins with the separation of small groups of authors that only work with people of their own groups from the larger interconnected network of collaborators. This is done by two magnets: one inactive that attracts the largest connected component and an active one that pulls the smaller components away from the rest. As seen in Fig. 9, the grouping of the smaller components and their placement away from the largest one makes it instantly easier to distinguish them from the rest of the graph, which “unwraps” horizontally.

The next action was making the CG group more distinguishable from the rest of the graph. This was done by placing a magnet that attracts it in a relatively central position (Fig. 10), making more evident not only the group itself, but also the publications and authors that link it to the rest of the graph. The attribute used to attract these nodes are the authors’ names. In this image it is also possible to see that the CG group serves as the only link between the graph and a faculty member and his or hers small network of collaborators.

With the focus now on the CG group, we isolated the common collaborators of two professors. We did this by first creating a magnet with requirements for the nodes that have path length of 1 to the two professors (Fig. 11a), and then applying a boundary shape on the same magnet to isolate the nodes we were looking for (Fig. 11b).

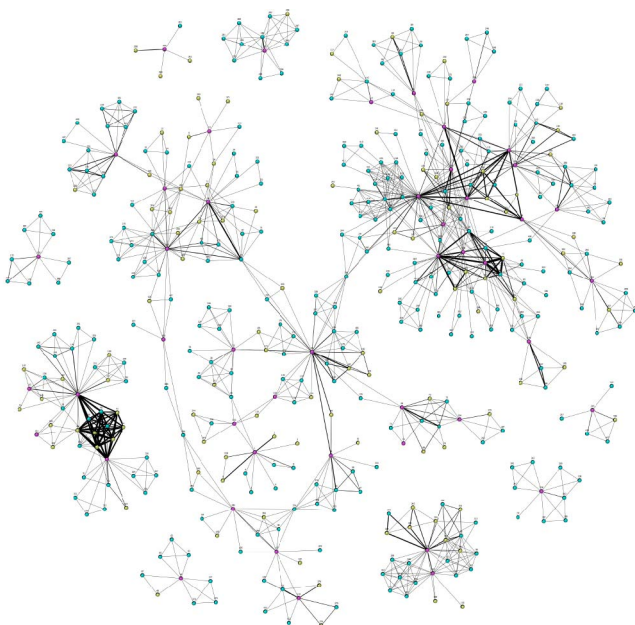


Fig. 8. The 2009 UFRGS Informatics Institute coauthorship graph.

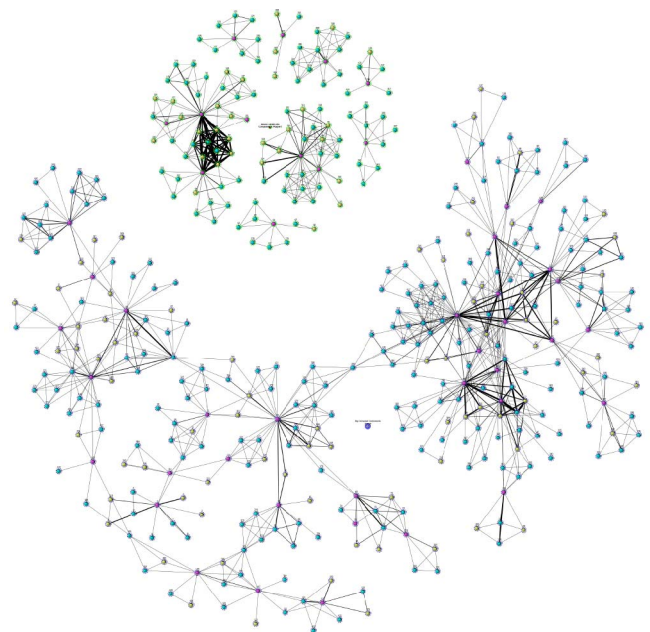


Fig. 9. The green magnet attracts the small connected components while the dark blue magnet highlights the largest one.

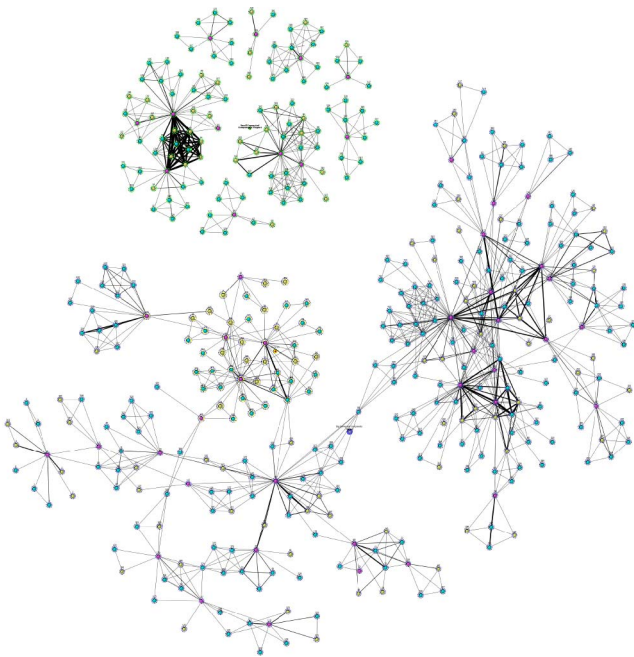


Fig. 10. Making the CG group stand out with a magnet in yellow.

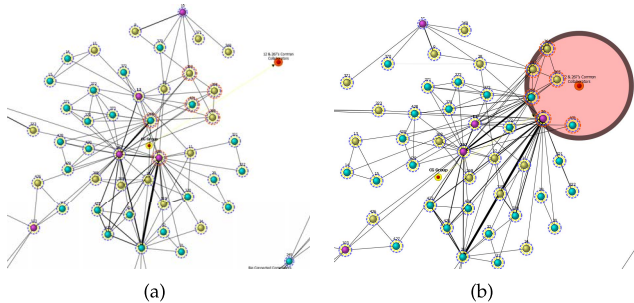


Fig. 11. Zoom on the CG group: (a) red magnet attracts common collaborators of professors 12 and 267 and (b) a boundary shape is added to it.

At this time, in the usage scenario, one can choose to move back from the CG group, and explore the larger component. The layout was then modified to emphasize the authors that did not publish in conference proceedings in 2009. This was achieved by a magnet that attracted all nodes that had edges with attributes indicating that they represented papers published in conference proceedings (Fig. 12). This search was refined by adding a requirement that all attracted authors be faculty members, resulting in only one attracted node—a professor belonging to one of the small connected components (Fig. 13). This professor's connected component can be circled (using a boundary shape) to make it easier to visualize the result of the exploratory task.

## 5 VALIDATION AND EVALUATION

To allow for a proper evaluation of MagnetViz, it is important first to contextualize it as to where it fits as a visualization technique. We used the model for design and validation of visualizations proposed by Munzner [23]. This model splits the process of visualization design into four nested levels, with the output of an upstream level above

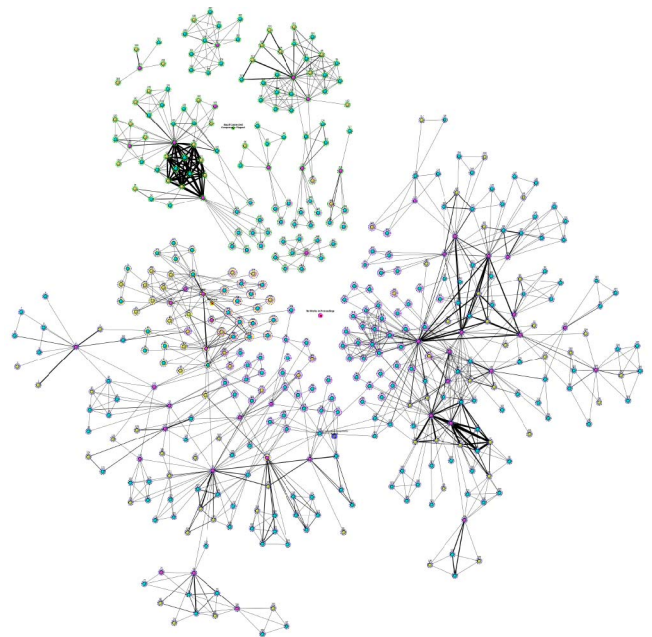


Fig. 12. Pink magnet attracts all authors who did not publish in conference proceedings.

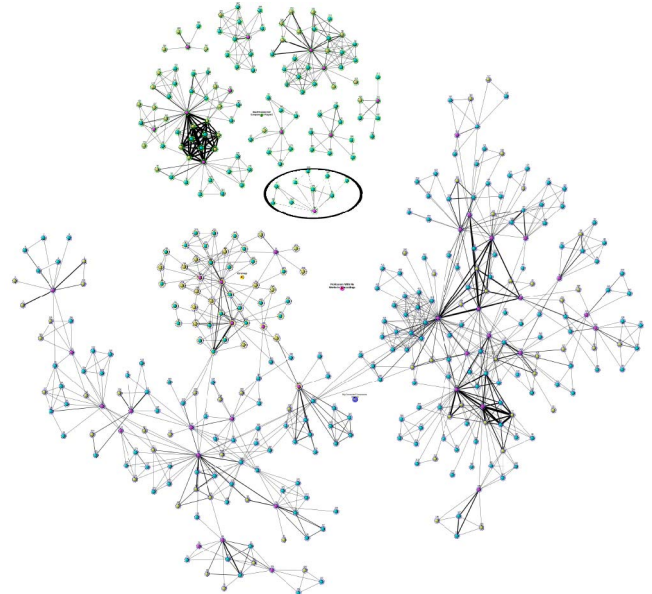


Fig. 13. Pink magnet attracts the sole faculty member who did not publish in conference proceedings.

being the input of the downstream level below it (meaning that an upstream decision—and possible error—will cascade to all downstream levels).

The topmost level of Munzner's model is the characterization of the problems and data of a particular application domain. This first level is about understanding the users' domain and establishing in this domain's own language what problems users have and want to solve. The next level is taking the requirements elicited in the first level and mapping them into generic operations into generic data types (e.g., from problems in social network analysis to operations on graphs). The third level consists of the design of how the data will be visually encoded and interacted with, that is, the design of the application itself.

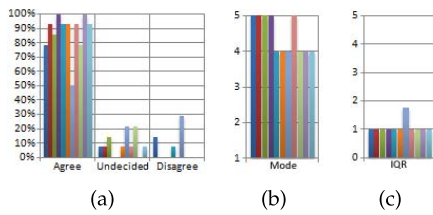


Fig. 14. S1—Identifying clusters.

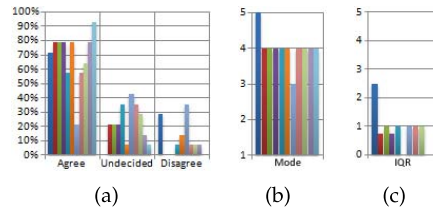


Fig. 15. S2—Identifying cut-nodes.

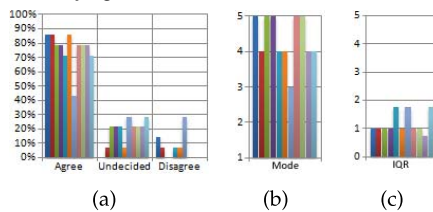


Fig. 16. S3—Identifying leaf nodes.

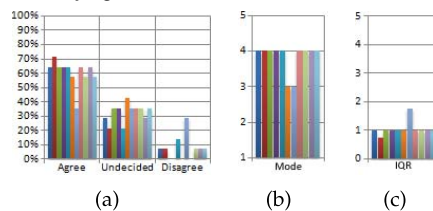


Fig. 17. S4—Identifying paths.

The final and innermost level is the creation of the algorithm that will be used to carry out the visual encoding and interaction designs of the previous level.

MagnetViz was designed to provide users with interactive tools that allow them to alter graph layouts in order to make them more meaningful. It can be thus classified as being on the third level of Munzner’s model. As such, it assumes that application designers have already successfully elicited requirements in the application domain and mapped them into operations on graphs. MagnetViz’s validation, then, is about ensuring that the visual abstraction and interaction scheme designed or chosen for the technique are effective. To do so it must be established: 1) how efficient the layouts produced with it are in conveying the information they were meant to communicate; 2) how clear its concepts are and how easy they are to learn and use; and 3) what advantages it has over related techniques. Steps 1) and 2) are covered in Sections 5.1 and 5.2, while step 3) is covered in the discussion presented in Section 6.

In order to decide whether to use MagnetViz or not in their applications, it is useful for designers to know which graph visualization tasks are supported by MagnetViz and how they can be performed using the technique. To aid them in this decision, in Section 5.3 we make a

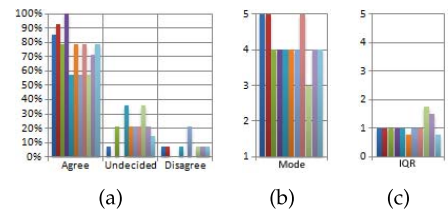


Fig. 18. S5—Layout clarity.

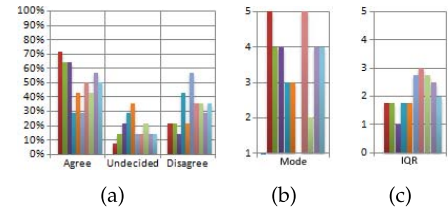


Fig. 19. S6—Clearer than the original.

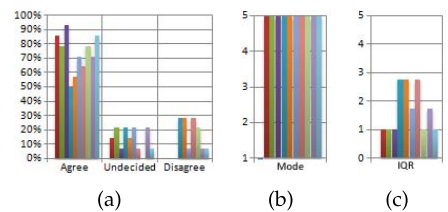


Fig. 20. S7—Conveys what is meant.

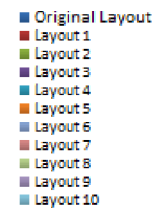


Fig. 21. Legend.

thorough assessment of MagnetViz against Lee et al.’s task taxonomy [7].

### 5.1 Quality of Generated Layouts

The evaluation of the layouts generated by MagnetViz was done with a user study. The tests were designed to prove the hypothesis that layouts achieved using MagnetViz properly emphasize semantically relevant information without suffering significant loss of aesthetic quality in relation to the original layout. As with the case study (Section 4), these tests were performed based on layouts built on the 2009 UFRGS Informatics Institute coauthorship network. This data set was used to provide a context that was of interest to the subjects, since many of them were represented as nodes in the graph.

#### 5.1.1 Subjects

Fourteen members of the CG group at UFRGS participated as subjects. They were between 21 and 34 years old and all had some previous knowledge of graphs and visualization.

#### 5.1.2 Procedure

The tests consisted of showing subjects an original force-directed layout of the 2009 coauthorship graph (Fig. 8), and 10 modified versions made with MagnetViz. Each

layout was accompanied by descriptions of what the image intended to portray (e.g., common collaborators of two members of the CG group, faculty members with no publications in journals, etc.). Subjects were asked to visually inspect each layout and give their level of agreement to seven positive statements on a five point Likert scale. Statements 1 to 4 concerned the ease of performing topology-based tasks on the graph (namely, identifying clusters, cut nodes, leaf nodes, and paths), while statement 5 was about the overall clarity of the layout. Statements 6 and 7 covered, respectively, clarity in comparison to the original layout and how successful they judged the layout to be in conveying the information it was meant to. These last two statements were not used with the original layout. Layouts shown included the five of the case study (Figs. 9, 10, 11, 12, and 13), as well as five others focusing on different aspects of the graph, such as the isolated groups of authors and the authors with the most collaborations and publications. Layouts modified with MagnetViz used all of its features, including boundary shapes, magnet hierarchy, and magnet intersection.

### 5.1.3 Results

For each statement, Figs. 14, 15, 16, 17, 18, 19, and 20 contain charts for the distribution of subject answers (a), mode (b) and interquartile range (c) for every layout. For the distribution charts, agreement answers to the Likert scale (“Strongly Agree” and “Agree”) were joined in a single category (“Agree”), with the same being done to disagreement answers (“Strongly Disagree” and “Disagree,” which became “Disagree”). A legend for all charts is provided in Fig. 21. As to the mode charts, answers were mapped to numbers, with 1 standing for “Strongly Disagree” and 5 being “Strongly Agree.”

As can be seen in Figs. 14, 15, 16, and 17, subjects mostly agreed with the topology statements. There was a non-negligible amount of subjects that marked “Undecided,” but very few disagreed (with the exception of layout 6). This is supported by the mode and IQR charts, which show that the mode was most often 4 (“Agree”), occasionally 5 (“Strongly Agree”), and rarely 3 (“Undecided”), never less. Variability was overall low, with the IQR being higher than 1 in only six cases (out of 44) and higher than 1.75 in only one case (2.50 for statement 2, original layout).

Layout clarity (Fig. 18) also fared well. For only two layouts (4 and 8) more than 21.43 percent of respondents marked “Undecided” and for only layout 6 more than 10 percent marked “Disagree.” Mode was below 8 only for layout 8, for which it was 3 (“Undecided”) and variability was overall low, with the IQR being equal to or less than 1 for all but layouts 8 and 9 (1.75 and 1.50, respectively).

In terms of the topology and clarity statements, the original layout fared similarly to the others and overall they received positive answers, supporting our hypothesis. From Fig. 19, from which it can be verified that while a slight majority of respondents agreed that the modified layouts were superior in clarity than the original, the answers were overall distributed among the three categories, providing more evidence that they were of similar quality to the original and further reinforcing the hypothesis. The overall

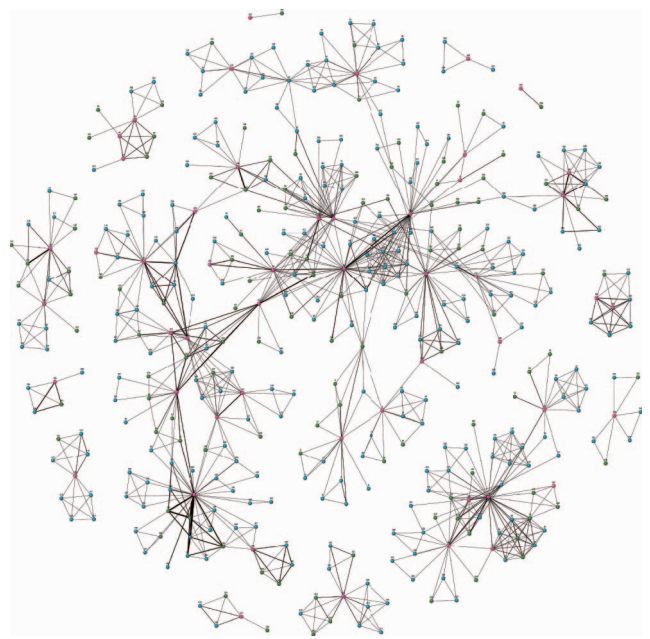


Fig. 22. 2007 UFRGS coauthorship graph. Pink nodes are faculty members, yellow are students, and blue are external collaborators.

aesthetic quality of the modified layouts is also supported by the mostly positive answers to the statement that the layouts successfully conveyed what they meant to (Fig. 20).

Although results shown here support our hypothesis, stronger evidence can still be obtained by doing more formal studies with more subjects and different graphs.

## 5.2 Evaluating MagnetViz

To evaluate MagnetViz as an interaction technique, a second user study was performed. With the exception of one subject who was not able to participate, subjects were the same as in the evaluation of layout quality (Section 5.1). This study was designed with the primary goal of answering the following questions:

- How well real users apprehended MagnetViz’s concepts?
- Are the concepts properly visually encoded?
- What is MagnetViz’s learning curve and how easy is it to use?
- How is MagnetViz actually used when put to test with real users?

To decrease the influence of the layouts seen during the previous study while maintaining subject interest in the data set, the tests here presented used the 2007 UFRGS coauthorship graph (Fig. 22).

### 5.2.1 Procedure

Initially, subjects underwent a training period in order to get acquainted with MagnetViz and its concepts. They were given several graphs of different sizes and told to experiment with the technique’s tools until they felt comfortable enough to move on to the actual test.

For the test itself, subjects were provided with the 2007 UFRGS Informatics Institute coauthorship graph (Fig. 22) along with a set of suggestions of what could be explored based on Lee et al.’s task taxonomy [7]. Subjects were told to

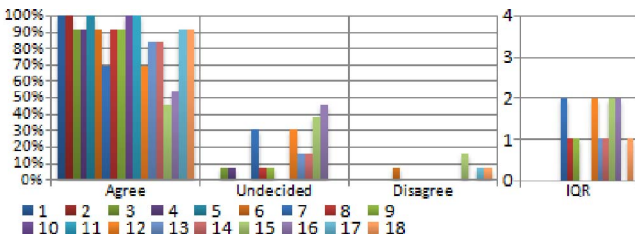


Fig. 23. Concepts.

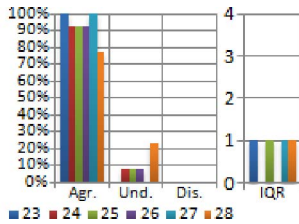


Fig. 24. Visual encoding.

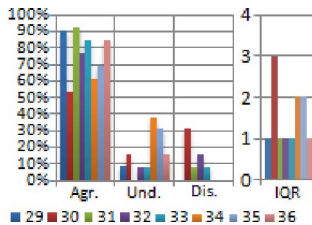


Fig. 25. Usability.

freely explore the graph, modifying the layout as they saw fit. For each created layout, they were told to generate an image file and provide a short description of what they were trying to visualize.

After trying out the prototype, subjects were told to answer a questionnaire concerning their experience with MagnetViz. The questionnaire was comprised of a set of 36 positive statements, to which subjects had to indicate their level of agreement on a five point Likert scale ranging from “Strongly Disagree” to “Strongly Agree.” The statements were divided into three categories. The first 18 statements addressed MagnetViz’s concepts, aiming at assessing whether subjects understood them and found them useful. The following four statements were about graph layout modification. Another six statements concerned the visual encoding the prototype uses for representing the concepts. The final eight were aimed at evaluating the prototype’s usability.

5.2.2 Results

As with the layout evaluation (Section 5.1), the answer distribution charts presented in Figs. 23, 24, and 25 summarize the Likert scale to three categories: “Agree,” “Undecided,” and “Disagree.” Mode is shown in numbers from 1 to 5, from “Strongly Disagree” to “Strongly Agree,” and a legend is provided for each chart, indicating which statement each color represents.

As can be seen in Fig. 23, the majority of respondents agreed to the statements concerning the MagnetViz concepts, with the mode for their answers being 5 (“Strongly Agree”) in every case. All subjects understood magnets and found them useful for layout modification (statements 1 and 2) and

a wide majority approved the criteria system (statements 3-9) and the boundary shapes, which were well understood (statement 10) and found to be useful to emphasize attracted nodes and modify the layout (statements 11 and 12). Magnet intersection was also both understood and found useful (statements 13 and 14), as was magnet hierarchy (statements 17 and 18). The only statements to which less than 70 percent of subjects agreed were the ones concerning the binding of nodes to a specific magnet in the case of an intersection (namely, statements 15 and 16). Interquartile range was equal to or below 1 for all statements but the ones for which more than 10 percent of respondents marked “Undecided” (namely, statements 7, 12, 15, and 16).

All subjects agreed to all statements concerning layout modification (statements 19-22, no chart provided). They found layout modification useful for emphasizing certain characteristics of the graph, and that it not only aids graph exploration but also brings to light new relations and motivates deeper exploration.

In terms of visual encoding (Fig. 24), over 90 percent of subjects found that magnets are adequately represented (statement 23), being easy to see which nodes attracted (statement 24) and which magnets attract a given node (statement 25). Over 90 percent of subjects also approved the use of magnet radius to encode attraction strength (statement 26) and how boundary shapes are represented (statement 27). While 76 percent found it easy to perceive the hierarchical relationships among magnets, 24 percent were undecided. There were no disagreements, with the answers mode being 5 (“Strongly Agree”) in statements 23, 24, and 27, and 4 (“Agree”) for all others. Variability was very low, as seen in the IQR chart.

As can be seen in Fig. 25, usability also fared well. Magnets and boundary shapes (statements 29 and 31) were found to be easy to use by more than 90 percent of subjects, but only 54 percent found it easy to alter magnet strength (statement 30). Creation and edition of criteria and their use for query building (statements 32 and 33) was also approved by more than 76 percent of respondents. Statements concerning more complex operations, such as binding and creation of child magnets (statements 34 and 35) received a modest agreement of between 60 and 70 percent of subjects, with the remainder being undecided. Finally, the overall ease use of MagnetViz for layout manipulation (statement 36) was agreed with by 85 percent of respondents. Mode was 5 (“Strongly Agree”) in statements 29, 30, 31, 34, and 25, and 4 (“Agree”) in all others. Variability was only higher than 1 in statements 30, 34, and 36.

From the images generated by the subjects and the comments they made in their descriptions of these images, we were able to verify how they actually used MagnetViz. Despite the fact that all subjects had access to some images generated by MagnetViz in the evaluation of layout quality, some of the layouts they created themselves were very different, often radically changing the original, which is shown in Fig. 22.

One of the approaches taken by the subjects was to use magnets to aggregate nodes that satisfy attribute-based criteria (such as all students of a given research group, all professors of that group, etc.) and see the relationships

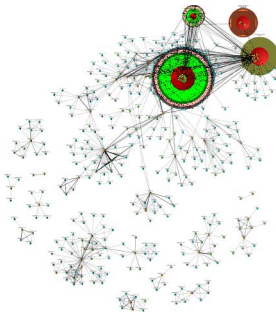


Fig. 26. User-made layout that separates the CG group and the different types of authors who published with them.

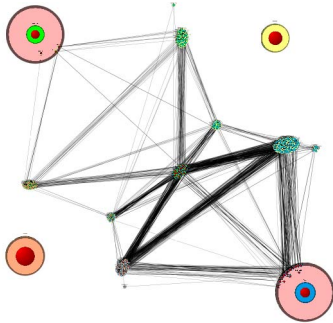


Fig. 27. User-made layout the relationships between nodes that satisfy different attribute-based criteria using magnet intersection.

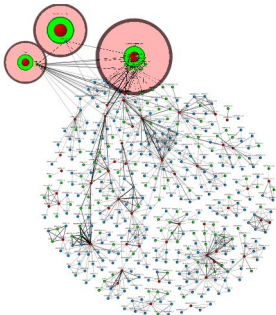


Fig. 28. User-made layout in which magnets are placed away from the graph to isolate the attracted nodes.

between the generated groups. Subjects that took this approach often used boundary shapes and very strong magnets, resulting in images such as the ones shown in Fig. 26. One particular subject used magnets to aggregate nodes that satisfied certain attribute-based criteria and their intersections to understand the relationships between the identified groups of nodes (Fig. 27).

Although subjects gave high marks to the usability of MagnetViz's tools, from the layouts they created we were able to observe that using it to create aesthetically pleasing layouts is not trivial. Subjects overall had a tendency to overuse boundary shapes and employ very strong magnets, often ending up with cluttered layouts that make relationships between individual nodes and the overall structure of the graph difficult to see (e.g., Fig. 26). Finding the ideal positions for the magnets on the scene also proved to be a challenge, with several subjects placing their magnets in empty regions near the borders of the workspace and away from the rest of the nodes in order to isolate their queries (Fig. 28).

While MagnetViz can be successfully used to create elegant layouts that emphasize certain information, as evidenced in Section 3.7 and the case study of Section 4, the results obtained with this experiment indicate that there is a learning curve before users are effectively able to do so. This is reinforced by our observation that after some time experimenting with the MagnetViz tools on the layout subjects started to make images more similar to the ones in the aforementioned sections.

To address some of the problems observed in user-made layouts, it would probably be interesting to make some changes in the technique, such as, for instance, placing a limit on the attraction strength of magnets and computing the size of boundary shapes automatically.

### 5.3 Task Taxonomy Assessment

Lee et al.'s task taxonomy for graph visualizations [7] was used in the assessment of MagnetViz since its first prototype [6] and follow-up improvements. To fully evaluate the concept of the technique in terms of which graph visualization tasks it supports, we present here the latest iteration of the task taxonomy assessment. To clarify our analysis we give examples of tasks within a coauthorship network context.

It should be noted that while MagnetViz supports 17 of the 23, depending on the context some can be performed more easily using other interaction techniques. MagnetViz was designed to work alongside other techniques and its appropriateness for a task is highly dependent on the context in which it is used.

#### 5.3.1 Low-Level Tasks

The taxonomy begins by defining 13 low-level visual analytic tasks, with 10 being supported by MagnetViz. Table 1 summarizes these tasks. Of the supported tasks, several are so by the very definition of magnets and criteria. **Retrieve Value** is accomplished by creating magnets with criteria set to attract nodes and edges that have specific values. **Determine Range** can be performed by building magnets with a combination of criteria that delimit a certain range of possible values. **Filter** and **Cluster** are performed by having all the nodes that are attracted by a magnet move toward it.

Some tasks are supported by specific criteria, while others rely also on features such as magnet intersection and hierarchy. **Find Extremum** can be performed by using criteria set directly to attract maximum or minimum values of a given attribute. **Find Adjacent Nodes** can be accomplished by a criterion acting on the path length of a node to another. **Set Operation** is inherently performed through the magnet criterion, as well as magnet hierarchy and intersection.

Other tasks are supported by the exploration of the data set through combinations of the technique's features, as well as visual inspection. Ideally, a visualization solution would include features such as labels to make their execution easier for the user. Among these, we find **Scan**, **Correlate**, and **Find Anomalies**. The first can be accomplished by seeing which nodes are attracted to a magnet, while the latter two can be achieved by combining the use of MagnetViz's features to sort through the data with visual inspection.

TABLE 1  
The 13 Low-Level Tasks of Lee et al.'s Task Taxonomy for Graph Visualizations

Task	Description	Example within a co-authorship network	Status
Retrieve Value	Given a specific set of cases, find attributes of those cases.	<i>In what conference was a certain paper published?</i>	S
Filter	Given some conditions on attribute values, find data cases satisfying those conditions.	<i>Which authors are faculty members?</i>	S
Compute Derived Value	Given a set of data cases, compute an aggregate numeric representation of those data cases, (e.g. average, median, and count).	<i>What is the average number of papers of all faculty members of a given institution in a certain year?</i>	U
Find Extremum	Find data cases possessing an extreme value of an attribute over its range within the data s.	<i>Which author has the most collaborators?</i>	S
Sort	Given a set of data cases, rank them according to some ordinal metric.	<i>Order authors by number of publications.</i>	U
Determine Range	Given a set of data cases, find the span of values within the set.	<i>Who published between 2004 and 2008?</i>	S
Characterize Distribution	Given a set of data cases and a quantitative attribute of interest, characterize the distribution of that attribute's values over the set.	<i>What is the distribution of the number of publications of authors?</i>	U
Find Anomalies	Identify any anomalies within a given set of data cases with respect to a given relationship or expectation.	<i>Is there a researcher who has not published?</i>	S
Cluster	Given a set of data cases, find clusters of similar attribute values.	<i>Which authors have published in a given conference?</i>	S
Correlate	Given a set of data cases and two attributes, determine useful relationships between the values of those attributes.	<i>Is there a connection between a number of publications and whether the author is a professor or a student?</i>	S
Find Adjacent Nodes	Given a node, find its adjacent nodes.	<i>What authors have collaborated with a given author?</i>	S
Scan	Quickly review the list of items.	<i>Which authors belong to a certain institution?</i>	S
Set Operation	Given multiple sets of nodes, perform set operations on them.	<i>Which authors have collaborated with two given authors?</i>	S

Tasks such as **Compute Derived Value** and **Characterize Distribution** are unsupported because they require analytic functions, which are not part of the technique. **Sort** is also unsupported, but could be made so by extending MagnetViz.

### 5.3.2 Topology-Based Tasks

Lee et al. [7] divided topology-based tasks in Adjacency, Accessibility, Common Connection, and Connectivity tasks. Two of the three **Adjacency** tasks are fully supported: "*Finding the set of nodes adjacent to a node*" is the same task as the low-level Find Adjacent Nodes, while "*Which node has a maximum number of adjacent nodes?*" is simply obtaining the node with the highest possible degree. Topology-based criteria allow both of these tasks, the first with a path length or neighbor criterion and the second with a degree criterion set to attract the node with the highest possible degree. The third task, "*Counting how many nodes are adjacent to a node,*" can be easily performed by simply showing a node's degree, for instance, by clicking on it.

All four **Accessibility** (direct or indirect connection) tasks are fully supported by creating magnets with path length criteria. These tasks are "*Find the set of nodes accessible from a node,*" "*How many nodes are accessible from a node?*," "*Find the set of nodes accessible from a node where the distance is less than or equal to n,*" and "*How many nodes are accessible from a node where the distance is less than or equal to n?*"

The **Common Connection** task ("*Given nodes, find a set of nodes that are connected to all of them*") is also fully supported by MagnetViz through different ways. It is possible, for example, to perform it by creating a magnet with path

length requirements to attract the nodes connected to each target node.

Of the **Connectivity** tasks, all are conceptually fully supported, also through different approaches. "*Find the shortest path between two nodes*" can be achieved by, for instance, using a topology-based criterion that attracts all nodes that are part of the shortest path between the two targets. "*Identify Clusters (subgraphs of connected components whose nodes have high connectivity)*" can be performed with the aid of a connected component criterion, with high density nodes being found by visual inspection or by placing a magnet with a degree criterion parented to the one that attracts the cluster. "*Identify maximally connected components*" can also be performed with a connected component criterion. Both "*Finding bridges*" and "*Finding articulation points*" can be performed with topology-based criteria or by using magnets to isolate nodes into groups that are highly connected or have common traits.

### 5.3.3 Attribute-Based Tasks

Attribute tasks are divided by Lee et al. [7] into those concerning nodes and those concerning edges. All four of these tasks are fully supported.

Of the tasks concerning nodes, "*Find the nodes having a specific attribute value*" can be performed by using the attribute-based criteria, while "*Review the set of nodes*" is basically the low-level Scan task. Of the tasks regarding edges, "*Given a node, find the nodes connected only by certain types of edges*" can be achieved by using a magnet with two requirements, one to ensure that the attracted nodes are neighbors of the given node and another to establish the

type of edge the nodes should have. “Which node is connected by an edge having the largest/smallest value” can be achieved by using a magnet set to attract the maximum/minimum value of a chosen attribute.

### 5.3.4 Browsing Tasks

Two Browsing tasks are described by Lee et al. [7] and although neither is directly supported by MagnetViz, our technique can still aid the user in performing them. For the task **Follow Path** (“Follow a given path”), for example, one can create a magnet that attracts the first node’s neighbors and then use other magnets to attract neighbors of the neighbors, and so on. As for **Revisit** (“Return to a previously visited node”), MagnetViz can aid through the visual reorganization of the graph provoked by the previous queries. Depending on what magnets the user inserted and where they were placed, the visualization might make it easier to go back to the first node and explore another path in its tree.

## 6 DISCUSSION AND FINAL COMMENTS

In this work we showed that, if properly employed, MagnetViz can be used to produce aesthetically pleasing layouts that better reflect semantically relevant information than simple force-directed layouts. From the graph’s topology to its metainformation, any characteristic users find to be important to their specific necessities can be used to generate layouts. As such, MagnetViz can be said to be a semantics-based technique.

Taking into account anything users find to be semantically relevant and being based on the modification of a topology-based layout, MagnetViz has advantages over both topology and attribute-based techniques. In contrast to attribute-based techniques, MagnetViz does not discard the graph’s structural information and uses a visual representation that is already very familiar to users (node-link diagrams laid out with force-directed algorithms). In contrast to typical topology-based visualizations, MagnetViz allows users to perform a large-scale modification of the layout based on both the graph’s topology and metainformation.

MagnetViz also has advantages over other techniques that modify graph layouts. The query system used for criteria composition allows magnets to be used to exert substantial semantically relevant change on a graph’s layout. This can bring about much more meaningful change than simply moving nodes around or fixing certain them to specific positions.

Like any technique, though, MagnetViz also has its drawbacks. One of its main disadvantages is that the quality of the layouts that can be made with it depends as much on users as on the technique itself. Since MagnetViz can be used for both layout creation and exploration, a lot of flexibility is required in how a layout can be modified. This makes it so that just as the technique can be used to generate meaningful, clean layouts, it can also be used to produce misleading, cluttered visualizations—a boundary shape, for instance, can either make a visualization clearer by isolating attracted nodes, or cause clutter if it is too small for the nodes it contains. In addition, the user studies have shown that

MagnetViz has a significant learning curve before users are able to generate aesthetically pleasing layouts.

As for the future of MagnetViz, several improvements are planned. Some are changes intended to address the issues found in the evaluation presented in this paper, and others are entirely new features designed to make the technique more powerful. Among the first type of improvements are the establishment of limits to the attraction strength to keep users from creating magnets that are too strong, and the automatic definition of boundary shape radius based on the number of attracted nodes, to avoid clutter.

New features planned include the ability of applying different layout algorithms to the nodes attracted by a magnet, in an approach similar to McGuffin and Jurisica’s [17]; weighted attraction strength based on the values nodes may have for numerical attributes; and attribute-based node aggregation, in a manner similar to PivotGraph [9]. Taking up on users’ suggestions, we also plan a filtering system that visually hides nodes and edges but continues to use them in the computation of the layout.

## ACKNOWLEDGMENTS

We thank our colleagues at UFRGS for serving as subjects in the user studies, and the anonymous reviewers for the insightful comments. We also thank CNPq for financial support.

## REFERENCES

- [1] G.D. Battista, P. Eades, R. Tamassia, and I.G. Tollis, *Graph Drawing*. Prentice Hall, 1999.
- [2] I. Herman, I.C. Society, G. Melancon, and M.S. Marshall, “Graph Visualization and Navigation in Information Visualization: A Survey,” *IEEE Trans. Visualization and Computer Graphics*, vol. 6, no. 1, pp. 24-43, Jan.-Mar. 2000.
- [3] S. Wasserman, K. Faust, and D. Iacobucci, *Social Network Analysis: Methods and Applications (Structural Analysis in the Social Sciences)*. Cambridge Univ. Press, Nov. 1994.
- [4] B. Cheswick, H. Burch, and S. Branigan, “Mapping and Visualizing the Internet,” *Proc. USENIX Ann. Technical Conf.*, pp. 1-12, 2000.
- [5] N. Henry, J.-D. Fekete, and M. McGuffin, “Nodetrix: Hybrid Representation for Analyzing Social Networks,” *ArXiv E-prints*, vol. 705, May 2007.
- [6] A.S. Spritzer and C.M.D.S. Freitas, “A Physics-Based Approach for Interactive Manipulation of Graph Visualizations,” *AVI ’08: Proc. Working Conf. Advanced Visual Interfaces*. pp. 271-278, 2008.
- [7] B. Lee, C. Plaisant, C.S. Parr, J.-D. Fekete, and N. Henry, “Task Taxonomy for Graph Visualization,” *BELIV ’06: Proc. AVI Workshop Beyond Time and Errors*. pp. 1-5, 2006.
- [8] T. Pattison, R. Vernik, and M. Phillips, “Information Visualisation Using Composable Layouts and Visual Sets,” *Proc. Asia-Pacific Symp. Information Visualisation*, pp. 1-10, 2001.
- [9] M. Wattenberg, “Visual Exploration of Multivariate Graphs,” *CHI ’06: Proc. SIGCHI Conf. Human Factors in Computing Systems*, pp. 811-819, 2006.
- [10] B. Shneiderman and A. Aris, “Network Visualization by Semantic Substrates,” *IEEE Trans. Visualization and Computer Graphics*, vol. 12, no. 5, pp. 733-740, Sept./Oct. 2006.
- [11] A. Aris and B. Shneiderman, “Designing Semantic Substrates for Visual Network Exploration,” *Information Visualization*, vol. 6, pp. 281-300, Dec. 2007.
- [12] A. Bezerianos, F. Chevalier, P. Dragicevic, N. Elmqvist, and J.-D. Fekete, “Graphdice: A System for Exploring Multivariate Social Networks,” *Computer Graphics Forum*, vol. 29, no. 3, pp. 863-872, 2010.

- [13] A.J. Pretorius and J.J. van Wijk, "Visual Analysis of Multivariate State Transition Graphs," *IEEE Trans. Visualization and Computer Graphics*, vol. 12, no. 5, pp. 685-692, Sept./Oct. 2006.
- [14] A.J. Pretorius and J.J. van Wijk, "Visual Inspection of Multivariate Graphs," *EUROVIS '08: Proc. Eurographics/IEEE-VGTC Symp. Visualization*, pp. 967-974, 2008.
- [15] D. Archambault, T. Munzner, and D. Auber, "Grouseflocks: Steerable Exploration of Graph Hierarchy Space," *IEEE Trans. Visualization and Computer Graphics*, vol. 14, no. 4, pp. 900-913, July/Aug. 2008.
- [16] S. Zhao, M. McGuffin, and M. Chignell, "Elastic Hierarchies: Combining Treemaps and Node-Link Diagrams," *Proc. IEEE Symp. Information Visualization (INFOVIS '05)*, pp. 57-64, Oct. 2005.
- [17] M.J. McGuffin and I. Jurisica, "Interaction Techniques for Selecting and Manipulating Subgraphs in Network Visualizations," *IEEE Trans. Visualization and Computer Graphics*, vol. 15, no. 6, pp. 937-944, Nov./Dec. 2009.
- [18] D. Archambault, H.c. Purchase, and B. Pinaud, "The Readability of Path-Preserving Clusterings of Graphs," *Computer Graphics Forum*, vol. 29, no. 3, pp. 1173-1182, 2010.
- [19] J.S. Yi, R. Melton, J. Stasko, and J.A. Jacko, "Dust & Magnet: Multivariate Information Visualization Using a Magnet Metaphor," *Information Visualization*, vol. 4, pp. 239-256, Oct. 2005.
- [20] D. Tunkelang, "A Numerical Optimization Approach to General Graph Drawing," PhD dissertation, School of Computer Science, Carnegie Mellon Univ., 1999.
- [21] T.M.J. Fruchterman and E.M. Reingold, "Graph Drawing by Force-Directed Placement," *Software Practice and Experience*, vol. 21, no. 11, pp. 1129-1164, 1991.
- [22] L. Verlet, "Computer 'Experiments' on Classical Fluids. I. Thermodynamical Properties of Lennard-Jones Molecules," *Physical Rev.*, vol. 159, no. 1, pp. 98-103, July 1967.
- [23] T. Munzner, "A Nested Model for Visualization Design and Validation," *IEEE Trans. Visualization and Computer Graphics*, vol. 15, no. 6, pp. 921-928, Nov./Dec. 2009.



**Andre Suslik Spritzer** received the BSc and MSc degrees in computer science from UFRGS, where he is working toward the PhD degree. His research interests range from computer graphics to the development and evaluation of visualization and interaction techniques in 2D and 3D.



**Carla Maria Dal Sasso Freitas** received the MSc and PhD degrees in computer science from UFRGS, where she has been a full professor since 1980. Her research interests range from the development of novel visualization techniques as well as the evaluation of 2D and 3D interaction in visualization applications.

► **For more information on this or any other computing topic, please visit our Digital Library at [www.computer.org/publications/dlib](http://www.computer.org/publications/dlib).**