Interactive Poster: CGV – Coordinated Graph Visualization

James Abello*
DIMACS
Rutgers University

Hans-Jörg Schulz†
Institute for Computer Science
University of Rostock

Heidrun Schumann‡
Institute for Computer Science
University of Rostock

Christian Tominski§
Institute for Computer Science
University of Rostock

ABSTRACT

Visualizing large hierarchized graphs is such a challenging task that it is hardly possible to represent all relevant information within a single comprehensible image.

To address that challenge, we pursue multiple linked view visualization. We report on a visualization framework for exploring hierarchized graphs that focusses on a modular architecture based on the model-view-controller (MVC) concept. MVC helps us in coordinating views as users explore and navigate the data. Several interaction facilities further support users in applying the framework for their data and their task at hand.

Keywords: Graph visualization, model-view-controller, interaction, lenses, dynamic filtering.

1 INTRODUCTION

Visual analysis of graph structures is a hot topic in many domains, as for instance system biology, network security, or sociology. In many of these domains, we are facing graphs that are too large and complex to be represented by a single visualization. In such cases, creating a graph hierarchy helps to drive interactive visual analysis [2]. The challenge here is that a graph hierarchy imposes certain requirements on the visualization. In our view, it is mandatory to represent the following items:

- clustering hierarchy H,
- anti-chains (abstractions of the underlying graph G), and
- data attributes associated to nodes and edges.

It is hardly possible to visualize all these aspects with a single technique. As demonstrated in previous work, multiple views are a solution to this problem [1]. So far, hard-wired view composition has been the chosen approach. We advocate here the use of an architecture where a more flexible set of views can be easily integrated and customized (e.g., to address particular needs of an application scenario). We report on how the model-view-controller pattern (MVC) can be utilized to achieve a flexible system architecture for interactive graph visualization.

2 FRAMEWORK ARCHITECTURE

The model-view-controller (MVC) concept is widely applied in scenarios where data (M) needs to be represented (V) and are subject to interaction (C). The strict separation of data, views, and interaction is the major benefit of MVC. Architectural separation of views and interaction does not necessarily mean that users are not allowed to interact with the views directly. Adhering to MVC allows for flexible architectures where components can be plugged in on demand. In particular, we instantiate MVC as follows.

The main data structure (M) contains all necessary elements to model a graph hierarchy (see [1]). It integrates the underlying graph G, the clustering hierarchy H, and associated anti-chains (i.e., abstractions of G) with their corresponding nodes and edges. It also takes into account (multivariate) data associated with nodes.

A view (V) is responsible for representing the data model. What aspects of the data are represented depends on the specific implementation of a view. We explicitly incorporate multiple views in order to represent different data aspects. In Sect. 3, we give examples of views currently available in the system.

Views are not allowed to modify the data (read-only access). However, the interactive exploration of a graph via anti-chains requires manipulation of the data model, in order to switch between different levels of abstraction. To guarantee that the data model is consistent at all times, even when represented by multiple views, manipulation requests are propagated to a controller (C). In turn, the controller notifies all views of the pending manipulation. If no view has an objection against the manipulation, the data model is altered. Then all views are notified about the change. It turned out to be a good solution to first notify the view that issued the manipulation request and then inform all other views. That gives users immediate feedback at the place where they performed the interaction. An overview of available interaction techniques is given in Sect. 4.

3 VIEWS IN CGV

MVC lays the ground for multiple view visualization. Multiple views help in communicating different data aspects and in performing different visualization tasks. This assumes that all views are arranged in a way that really supports the task at hand. Since the preferred arrangement depends on users and their tasks, we embed the views in a docking framework. This allows users to create arrangements different from the default setting, and furthermore, to store and reload customized arrangements.

Based on the arrangement used in [1], we designed the default setup of the framework as shown in Fig. 1. In particular, we provide the following views:

- *Graph view* provides at all times a node-link view of the current visible anti-chain, where nodes and edges are aggregations of the underlying graph. The purpose of this main view is to visualize structure and clusters, and to highlight a selected data attribute by means of color. The graph layout can be generated by a variety of algorithms, as for instance LinLog or a classic spring embedder.
- *Textual tree view* also visualizes the current anti-chain, however the represented edges are edges of the clustering hierarchy H. This view's purpose is to show the structure of H, labels, and to drive easy navigation. Enhanced visual and interaction capabilities support the user [4].
- *Graphical hierarchy view* gives an overview of the entire clustering hierarchy H. Additionally, a user-chosen data attribute can be color-coded.
- *Matrix view* provides a density visualization of the macrograph determined by the current anti-chain. This view enables users to spot dense graph clusters that may be used as triggers for further exploration.

*e-mail: abello@dimacs.rutgers.edu
†e-mail: hjschulz@informatik.uni-rostock.de
‡e-mail: schumann@informatik.uni-rostock.de
§e-mail: ct@informatik.uni-rostock.de

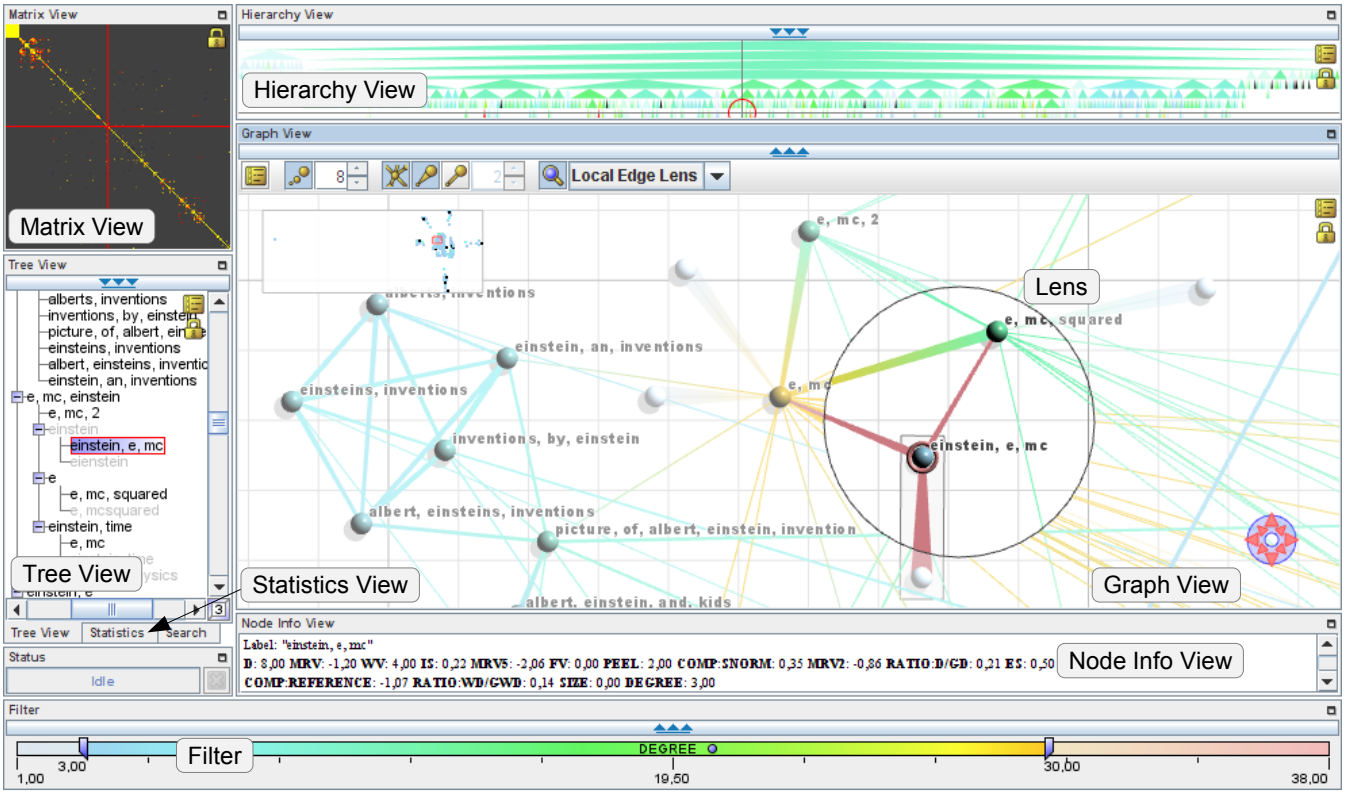

Figure 1: CGV – Coordinated graph visualization with multiple views.

- *Statistics view* represents meta information (e.g., size of the current anti-chain or the number of nodes and edges currently visible) in textual form, which is very helpful when exploring an unknown data set (not visible in Fig. 1).
- *Node Info View* represents meta information for the currently focussed node.

4 Interaction Facilities

To facilitate easy data exploration, we provide common interaction techniques, such as zoom&pan or scrolling. To allow users to switch between different abstractions of the hierarchized graph, nodes can be expanded and collapsed. Additionally, users can select and focus on nodes. All views follow consistent common interaction policies (e.g., double left click on a node in any view will result in expansion of that node). Recall that views do not alter the data model, but propagate requests to the controller. This decouples the specific physical action (e.g., click) from the effect (e.g., node expansion) and enables us to provide a basic undo/redo mechanism.

Beyond the aforementioned interactions, the system offers several novel interaction facilities:

- *Edge-based navigation* aims at supporting the common task of path navigation. For that purpose, we make the edges of a focussed node interactive. Clicking on such an edge navigates to the respective neighbor and then focusses on that node automatically.
- *Lenses* are helpful tools to support locally restricted visualization tasks [4]. They can help to tidy up edge clutter (task: "Which edges connect to a node?"), or can perform local transformations on the graph layout to bring possibly distributed nodes of interest close together (task: "What are the neighbors of a node?").
- *Dynamic filtering* is provided to support users in exploring the data for interesting nodes or selecting nodes relevant for a particular task at hand. Our filtering mechanism allows for logical combinations of basic filters, which operate on node attributes (e.g., quantitative values or categorical labels). We utilize a sieve metaphor to ease the interactive composition of filters. Through automatic fading or omitting filtered nodes, users get immediate visual feedback on the filtering result.

5 Conclusion

This work illustrates the use of the model-view-controller pattern (MVC) for coordinated graph visualization. This is exemplified in a system termed CGV (available at [3]). The proposed framework offers several common linked views and some novel interaction facilities (lenses and edge navigation). We are currently exploring the usefulness of other visualization techniques to be integrated into the framework. This is work in progress towards the end goal of facilitating user- and task-dependent setups (i.e., selection of techniques and their screen arrangement).

References

[1] Abello, van Ham, and Krishnan. ASK-GraphView: A Large Scale Graph Visualization System. *IEEE Transactions on Visualization and Computer Graphics*, 12(5), 2006.

[2] Herman, Melançon, and Marshall. Graph Visualization and Navigation in Information Visualization: A Survey. *IEEE Transactions on Visualization and Computer Graphics*, 6(1), 2000.

[3] Tominski. CGV prototype. `http://www.informatik.uni-rostock.de/~ct/CGV/CGV.html` (accessed June 2007).

[4] Tominski, Abello, van Ham, and Schumann. Fisheye Treeviews and Lenses for Graph Visualization. In *Proc. IV'06*, London, 2006.