Interactive Visualization of Large Bipartite Networks Assisted by Multilevel Strategies

Renato Fabbri[0000−0002−9699−629X], Alan Valejo[0000−0002−9046−9499], and Alneu A. Lopes[0000−0003−3112−4746] Maria Cristina F. Oliveira[0000−0002−4729−5104]

University of São Paulo, São Carlos SP, BR
renato.fabbri@gmail.com, alanvalejo@gmail.com, alneu@icmc.usp.br
cristina@icmc.usp.br

Abstract. Bipartite, or two-layer, networks are pervasive in modeling real-world phenomena and play a fundamental role in graph theory. Multilevel methods, introduced for solving optimization problems on networks, have been applied to the problem of drawing simple ("unipartite") networks, but not for interactive visualization of bipartite networks. In this work, we present a proof-of-concept implementation on the use of the multilevel method for this purpose. We developed a web-based visualization interface in which multilevel coarsening algorithms are applied to obtain a hierarchy of gradually simplified representations of an input bipartite network. Node-link representations of the resulting network hierarchy can be presented to users following a genuine route of the "overview first, zoom and filter, details on demand" visual information seeking mantra, as analysts may depart from a coarser representation and select super-nodes or sub-graphs (network partitions) to be expanded and shown at greater detail. Such a solution allows for interactive and intuitive navigation of large-scale network structures and can provide the basis for more elaborate visual mappings of network data sets.

Keywords: Network visualization · Bipartite networks · Multilevel strategies · Big data · Complex networks · Data visualization

1 Introduction

The visualization of large-scale networks poses challenges in terms of both the computational cost and the effective presentation of information [30, 29]. These issues may be aggravated in the case of bipartite networks, due to their sparsity and topological complexity. In bipartite networks the set of nodes is split into two partitions called "layers" and links are not allowed between nodes in the same partition. Such networks arise often and naturally from the representation of relations between entities of two kinds, e.g. documents and terms, papers and authors [23, 21, 13], or patients and genes [20]. Furthermore, real-world networks are often bipartite and most unipartite networks are actually projections of bipartite networks or exhibit bipartite properties [14, 15].

Multilevel strategies (see Section 2) perform an incremental coarsening of an original network to yield a sequence of gradually simplified representations, i.e. with fewer nodes and links. They have been traditionally employed to enable executing expensive algorithms on large-scale networks[1]: the rationale is to run the algorithm on a coarsened version that preserves the major properties of the input network to compute an initial solution, which is projected into the inverse hierarchy of coarsened networks to yield a solution to the problem relative to the original network [32, 22]. The method has been mostly applied to identify community structures in large networks [26], but also to other problems, including computing node-link layouts of simple (i.e. "unipartite") networks [17, 34, 19, 16, 12, 3, 31, 5, 7, 18]. We are aware of a single contribution (yet unpublished) in the context of bipartite networks, in which a visual metaphor is devised to assist users executing multilevel methods [10]. In general, solutions for node-link visualizations of large networks often rely on aggregation of nodes in clusters or communities to reduce visual clutter and create simpler representations amenable to user interaction [1, 4, 35, 8, 11, 25].

In this paper we present a system that employs the multilevel method to enable interactive visualization of large bipartite networks with node-link representations. Its underlying rationale is to present first a coarser instance of the network with which a user can interact to request for super-nodes (or groups of them) to be uncoarsened and visualized in more detail. The resulting interface enables interactive navigation on large networks, e.g. by gradual and on-demand uncoarsening of super-nodes, combined with functionalities for zooming and requesting complementary information on nodes and links. Besides assisting in the analysis of data represented as bipartite networks, the visualization may assist developers of multilevel algorithms, e.g., for inspecting and comparing results yielded by different algorithm and parameter choices. We consider the current implementation as a proof-of-concept to demonstrate the feasibility of using the multilevel method to support effective visualization of large bipartite networks using node-link views.

2 A brief description of the multilevel method

In this section we address two topics. First we introduce some fundamental concepts and present an overview of the multilevel method in the context of bipartite networks. Then we outline how it can be instantiated to support interactive navigation on such networks.

2.1 Multilevel coarsening of bipartite networks

A bipartite network $G = (V, E)$ consists in a set V of nodes which is partitioned in two subsets with no links between nodes in the same set, i.e. $\exists\, V_1, V_2 :\ V_1 \cup$

[1] By large-scale we mean networks with up to a few hundreds of thousands vertices, in contrast to massive-scale networks with millions to billions vertices.

$V_2 = V,\ V_1 \cap V_2 = \emptyset$, and $E \subseteq V_1 \times V_2$. The variable names are borrowed from traditional graph (network) theory, with Vertices (nodes) and Edges (links). One may regard the network as $G = (V, E, \sigma, \omega)$, with $\sigma : V \rightarrow \mathbb{R}^*$ and $\omega : E \rightarrow \mathbb{R}^*$, where $\sigma(v)$ is the weight of the node v and $\omega(u, v)$ is the weight of the link (u, v).

The multilevel optimization method is a meta-heuristics employed to guide, modify and potentially fix a solution obtained from a target algorithm. It operates in three phases, as illustrated in Figure 1, namely coarsening, solution finding and uncoarsening. Valejo et al. [32] introduced a general framework applicable to bipartite networks. They presented novel and efficient matching and coarsening algorithms and validated them in problems of community detection and dimensionality reduction. They also provided empirical evidence that the proposed coarsening algorithms preserve the essential topological features of an original input bipartite network. Here we delineate the stages of the multilevel method in the context of our bipartite network visualization contribution.

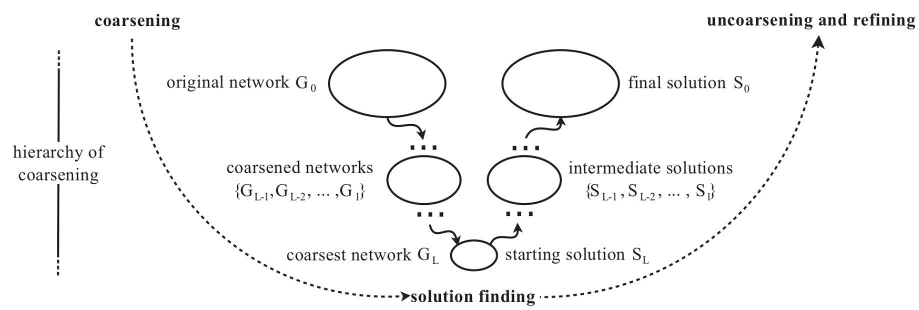


Fig. 1: Phases of the general multilevel optimization method. A sequence of incrementally reduced networks is obtained in the coarsening phase. An initial solution is found applying a target algorithm in the coarsest network instance in the solution finding phase. This solution is projected back to the initial, uncoarsened, network in the uncoarsening phase.

The coarsening phase creates a hierarchy of coarsened networks G_l, l integer $\in [0, L]$ where G_0 is the original network and $|V_i| \geq |V_{i+1}|$, resulting in a hierarchical representation of G_0 at decreasing levels of detail. The process is carried out by two algorithms: *matching*, which decides which nodes to merge, and *contracting*, which computes a reduced representation from the input network and the matching. In general, pairs of nodes are selected to be merged into super-nodes, and most often a matching M consists of a set of non-adjacent links, i.e. $\forall\ l_1, l_2 \in M, u \in l_1 \Rightarrow u \notin l_2$ [33, 9]. For bipartite networks, we rely on the matching algorithms introduced by Valejo et al. [32] in which the matching must satisfy two restrictions:

- A node may only match nodes on the same layer.

– A node may only match nodes that are reachable in two hops, i.e., via paths formed by two successive links (i.e. the closest possible nodes in the same layer).

The matching is followed by the contraction of the network into a coarser form. Typically, the nodes matched are merged into a super-node weighted according to the number of of its composing nodes, and the links incident to such nodes are joined into super-links, weighted based on the accumulated weights of the links merged. The solution finding phase usually involves executing a computationally expensive algorithm, which becomes feasible in the coarsest network. In the uncoarsening phase the solution found in the coarsest network instance G_L is projected back to the original, uncoarsened, network G_0. Uncoarsening is thus performed across the complete hierarchy, from G_L to G_0 with successive refinement of the solutions to avoid local minima and improve solution quality.

In applying the general multilevel framework to obtain an interactive visualization solution, some adaptations are in order. In this scenario the goal is to obtain a visual mapping of the network, i.e. the solution finding phase corresponds to computing a layout of G_L suitable for presentation. Uncoarsening is performed on-demand, through specific user requests to expand selected network super-nodes for more detailed visualization, which is possible displaying them at the subsequent hierarchical level. The procedure is delineated in the BiNetVis algorithm (Algorithm 1) and further detailed in Sections 2.2 and 3.

This solution avoids unnecessary complexity and, most importantly, avoids overloading the visualization with information beyond the cognitive convenience of the user and the computational power available to ensure real-time interactivity. Users can interact to specify the desired coarsening algorithm, select the node-link drawing algorithm, select the target super-nodes for uncoarsening, navigate the network, access data and metadata, tune the visual mapping options, for instance to modify colors and resize and move nodes.

2.2 The navigation pathway

The abundance of information within large-scale networks makes pertinent the application of the well-known "visual information-seeking mantra" [28]: *overview first, zoom and filter, then details-on-demand.* This mantra embeds a number of visual design guidelines, such as the details-on-demand interaction strategy, and provides a widely acknowledged framework for designing information visualization applications. The navigation pathway supported by the current implementation of the proposed solution is illustrated in Figure 2. Accordingly, exploration starts with an overview yielded by the visual mapping of the coarsest network instance, obtained using standard layout algorithms for node-link diagrams. The user may then zoom into specific network regions and request details using several operations, most importantly:

– Reposition nodes, expose linking patterns.

```
Input:
   bipartite network: G
   maximum number of levels: L ∈ [0, n] ⊂ ℤ
   reduction factor for each layer: rf_1, rf_2 ∈ (0, 0.5] ⊂ ℝ.
   layers to be coarsened: layers ∈ {1, 2}
   user command given through the visual interface: C
Output:
   Visual mapping of the network: V
i ← 1;
while i ≤ layers do
   l ← 1;
   while l ≤ L do
      M ← Matching_b(G_l, i, rf_1, rf_2);
      G_{l+1} ← Contracting_b(G_l, M);
      increment l;
   end
   increment i;
end
V ← map_to_screen(G_L);
while C is not 'exit' do
   C ← user command;
   V ← transform_visual_mapping(V, C);
end
```

Algorithm 1: ***BiNetVis*** algorithm for multilevel visualization of bipartite networks. The routines **Matching**$_b$ and **Contracting**$_b$ refer to any matching or contracting algorithms applicable to bipartite networks [32]. The routine **map_to_screen** may refer to any node-link layout algorithm followed by a rendering of the network to the screen. The user may then modify the visual mapping, e.g., request to uncoarsen selected super-nodes or use any of the other interaction commands available, such as request metadata, change node position, change node and link color or transparency, perform zoom and pan, etc., as described in Section 3.

– Request metadata, e.g. for a node one may ask for its degree, or how many predecessors it has in the previous level, or how many successors in the subsequent level.
– Adjust the visual mapping of network features, such as map the size of glyphs representing nodes to node degree or to the number of node predecessors, or modify link transparency.
– Request the uncoarsening (i.e., expansion to the upper level) of a selected super-node or group of super-nodes.

Other operations are convenient to adequate the visual mapping to the current user needs, e.g., setting the network size, expanding super-nodes, exposing levels and highlighting topological features. These operations are further detailed in Section 3.

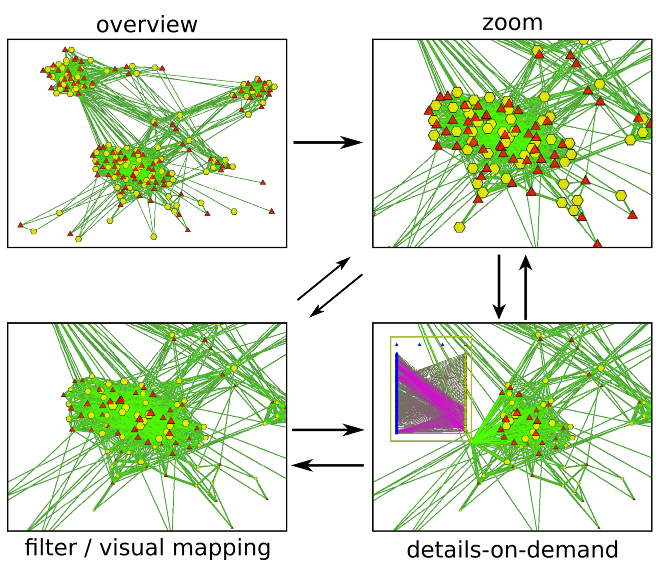


Fig. 2: The navigation pathway on bipartite networks made possible using the multilevel method coupled with auxiliary interaction tools. Compliant with the "visual information seeking mantra" [28], initially it displays a view of the bipartite network in its coarsest form. The user may then zoom-in into regions and select specific nodes or all nodes within a region to inspect and visualize in further detail, apply filters and modify the visual mappings of topological elements, or request to see the corresponding nodes in further detail, drilling-down to the next network level. These operations are user-driven and may be carried out in any sequence.

3 Software implementation

We implemented the framework described in Section 2 using scientific, database and web resources, making it available within a web page for use through simple mouse-driven actions and requiring no software installation beyond a web browser. The software has been named BiNetVis (from Multilevel Bipartite Network Visualization) and is exemplified in Figures 3, 4 and 5. In the following we describe its functionalities and the underlying technologies.

3.1 Using BiNetVis

In using BiNetVis, the analyst first uses a drop-down menu to select a network of interest, s/he may also upload a new network using the same menu. The next step is to select one of the available network layout algorithms and possibly tune the coarsening algorithm, as illustrated in Figure 4, although default settings are reasonable for a newcomer. Once the "render network" button is hit the network node-link representation is mapped to the screen according to the selected layout algorithm. Subsequent usage relies on manipulation of the visual mapping controls presented on the canvas by means of user actions and related mouse operations. These steps are illustrated in Figure 3.

The multilevel method yields a hierarchical representation of the network at decreasing levels of detail (as depicted in Figure 1), where the number of hierarchy levels is an input parameter to the coarsening algorithm. The proposed visualization initially shows a node-link representation of the the coarsest network G_L (the upper level instance) showing it at the minimum level of detail. The network G_L is thus the initial focus of user interaction. The focus can be directed to other levels of the multilevel hierarchy, but a single level can be the current focus to which any interaction actions will apply. Thus, there is always a user-selected level to which interface controls apply, such as link coloring or expanding super-nodes to reveal their predecessor nodes. As the reader may infer, the numbers of visible nodes and links in each level are not fixed throughout the navigation. The user can modify the focus level in the table to the right of the canvas, as shown in Figure 5, which also shows the number of visible and total nodes and links at each level. The user may also interact with the table to modify node color and shape or link color at any layer/level, as well as toggle show/hide links with left/right clicks on the corresponding colored cell.

The toolbar on top of the canvas in Figure 5 holds further controls organized in groups according to their target interface element. The first group of controls affect graphical attributes of nodes: size mapped to degree or to number of predecessors, reset size, modify transparency and rotation. The second group affects link properties: thickness, transparency, proportionality to weight using thickness or transparency, and reset proportionality.

The third group refers to controls more specific to how BiNetVis operates as a multilevel visualization. The first control returns information relative to a selected (clicked) node: id, level, number of predecessors at the upper level, successor id (if applicable), degree, strength (sum of link weights). The second

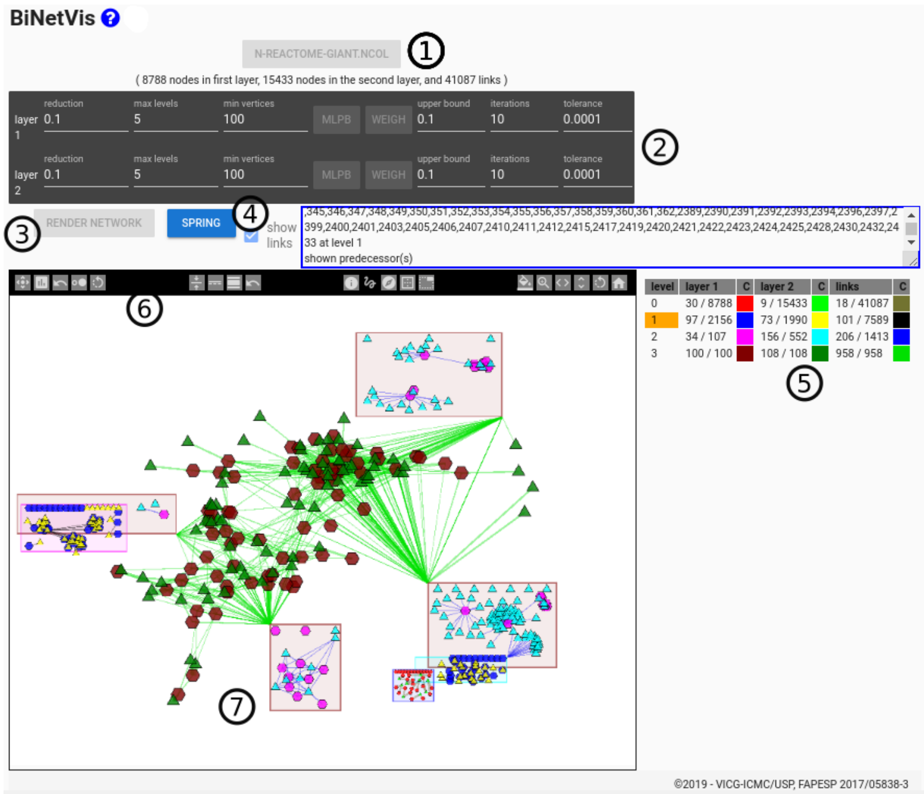


Fig. 3: The BiNetVis interface: (1) drop-down menu for selecting or uploading a network for analysis; (2) control widgets to select and parameterize the coarsening algorithm; (3) the render button to display the network; (4) auxiliary widgets for selecting the network layout, show/hide links, and area to display user requested information on nodes or links; (5) interactive table holding information about the multilevel hierarchy levels and the bipartite network layers; can also be used to set the visualization on the canvas; (6) toolbar with controls for navigation of the network and fine-tuning the visualization (detailed in Figure 9 of Appendix B); (7) canvas with the node-link representation of the multilevel network. Notice that nodes shown belong to four distinct hierarchical levels: level 3 (dark red/green), level 2 (pink/cyan), level 3 (navy blue/yellow) and level 2 (red/bright green). The current interaction focus in on level 2, as indicated by the orange mark.

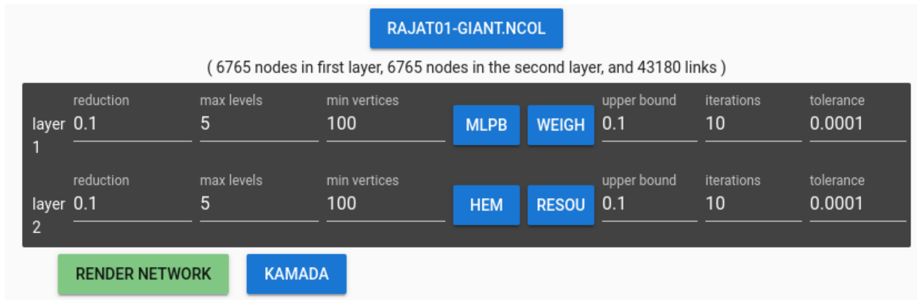


Fig. 4: The initial step of the BiNetVis usage is to apply the multilevel coarsening to an input network. The button at the top is for network uploading and selection. In the gray box, the user chooses the coarsening algorithm and sets its parameters (Appendix A details the parameters of coarsening algorithms). The blue button at the bottom is a drop-down menu where the user can select the node-link layout algorithm, where multiple alternatives are provided. The green button renders the network to the canvas and initializes further control widgets. By hitting the green button, except for the layout button all the control elements above are disabled as the interface enters the visualization mode.

control allows moving nodes: specific nodes are moved if clicked and dragged; a rectangular region can be defined (clicking on an empty region in the canvas and dragging then releasing the mouse) to move all nodes within. The third control allows expanding super-nodes to expose their predecessor nodes and links. Again, the user specifies a rectangular region and the super-nodes within are bounded within the region and replaced with their predecessor nodes, which exist at the upper hierarchical level (thus at a greater level of detail). Thus, an expanded super-node is identified by a bounding rectangle that contains a view of its predecessor nodes and links. The same control can be employed to join expanded super-nodes, but it was found convenient to include a fourth control to assist in this task, which allows joining two expanded super-nodes by clicking on them in sequence. The final control in this group supports three operations on expanded super-nodes: if an expanded super-node is clicked on and dragged its rectangular shape is resized accordingly; if the super-node is clicked on and released (no dragging) its links to the other super-nodes are shown attached to the next counterclockwise corner of the bounding rectangle. When the expanded super-node is resized the spatial layout of its child nodes is optimized to the modified drawing area. This is most useful in combination with the drag tool, to rearrange the nodes at the finer level of detail in a more compact disposition. Upon a super-node resize the child nodes are rearranged to fit the new rectangular area. The final set of controls act on the canvas: they allow changing the background color, zooming, left-right/up-down panning, rotating the entire network, and toggling between the current zoom and pan configuration and the initial settings.

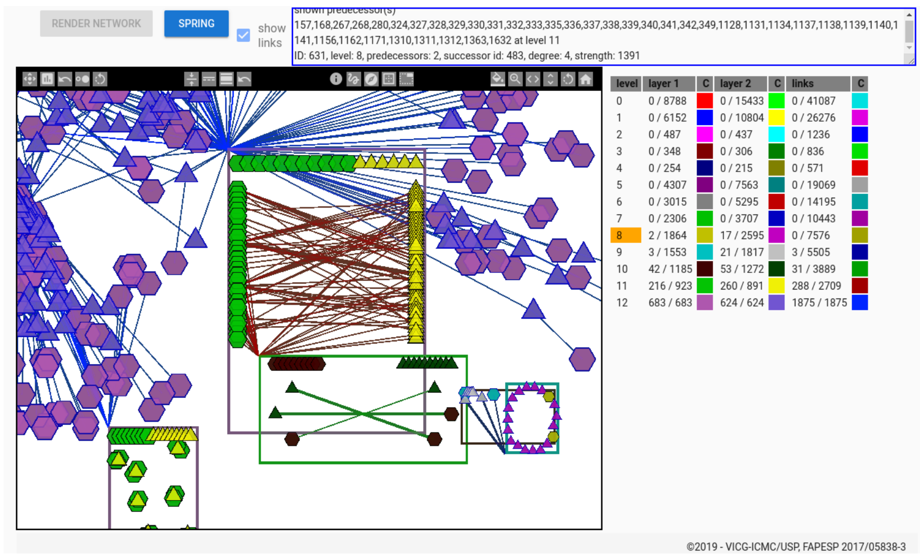


Fig. 5: Interface elements of BiNetVis in the visualization mode, where the user can navigate in the network and modify visualization settings. S/he may change the layout used when super-nodes are expanded and their predecessors rendered. The checkbox to the left of each level may be used to show or hide the corresponding links at the level. The text area displays user-requested information or information useful for navigation. The canvas holds the node-link representation of the network, possibly depicting nodes at multiple levels of the multilevel hierarchy. The node-link representation may be modified using the toolbar controls and the controls the table on the right, which also shows key information on the status of the visualization. Here the current focus is on level 8 from a multilevel network with 12 hierarchical levels.

There are many subtleties on the orchestrated use of these controls. As a facilitated means to convey the navigation possibilities, the user is invited to watch a demo video[2] and use the BiNetVis interface[3]. Furthermore, a simple case study is described in Appendix A. The more seasoned software developers may browse the code, upload a local instance of the software, and make changes to its source code[4].

3.2 Underlying software technologies

Components of BiNetVis are mostly written in a combination of JavaScript and Python: it uses Vue.js (set up by Nuxt.js) in the front-end client, the back-end is a Flask Python server, used to perform specialized or heavy calculations. A secondary server, a FeatherJS, is used to facilitate contact with the database and real-time multi-user interaction. The data is stored in a MongoDB database and ordinarily in the file system, while the multi-user interaction is deactivated for now to avoid unnecessary complexity. Multiple coarsening algorithms for bipartite networks available from a previous implementation [32] are accessed by the Flask server. The fast WebGL 2D rendering on the canvas is performed using Pixi.js.

In order to comply with the goal of handling large bipartite networks, the network geometry on the canvas uses only triangles as primitives (for nodes) and straight lines (for links). The user may wish to further alleviate the computational burden by not rendering the links. To emphasize the bipartite nature of the network and render the visualization more aesthetically compelling, the user may modify the node shapes in one (or both) layers to hexagons, instead of triangles. Using these features, and within the technologies described, we have visualized networks with tens of thousands of nodes without perceiving any lag in the interactive navigation and transformations, with links shown, even when running the system on ordinary machines, e.g. with 8GB RAM DDR3, a first generation i7 processor and a 1GB GPU.

4 Conclusions and further work

We introduced a visualization interface for bipartite networks assisted by the multilevel method that admits a conceptual organization very consistent with the well-known visual information seeking mantra stated as *overview first, zoom and filter, then details-on-demand*. As exemplified in the software description, it allows users to obtain an overview of the major topological structures in a network and then focus on relevant elements for which further details can be displayed. The combination of the multilevel strategy with suitable software technologies and computationally inexpensive design decisions regarding the rendering of node-link representations yields a visual interface manageable with

[2] http://rfabbri.vicg.icmc.usp.br:3000/multilevel2/about
[3] http://rfabbri.vicg.icmc.usp.br:3000/multilevel2/topdown
[4] https://github.com/ttm/netText

simple interactive operations that can effectively display large-scale networks. In summary, this software is introduced as a proof-of-concept on the feasibility of employing multilevel strategies for using the familiar node-link views to visualize large bipartite networks. The same underlying rationale is applicable to unipartite and to heterogeneous networks as long as the underlying multilevel methods are provided.

The proposed visualization solution could be incorporated into a visual analytics environment for large networked datasets, with added tools to enable data analytics. It does require further validation on practical analytical settings, as the usage scenarios of interactive knowledge discovery are inherently complex. We believe that tuning this implementation into a tool to guide and inform developers of novel multilevel algorithms and applications would also be an interesting development. For instance, certain multilevel strategies require selecting pivot nodes to guide the coarsening procedure, a task that could benefit from an interactive visual interface. The same applies to developers who wish to compare the outcome of multiple executions of coarsening algorithms, e.g., with different parameter settings.

Acknowledgements The authors acknowledge the financial support of the São Paulo State Research Foundation (FAPESP grant 2017/05838-3) and the National Council for Scientific and Technological Development (CNPq, grant 301847/2017-7). The views expressed do not reflect the official policy or position of FAPESP or CNPq.

References

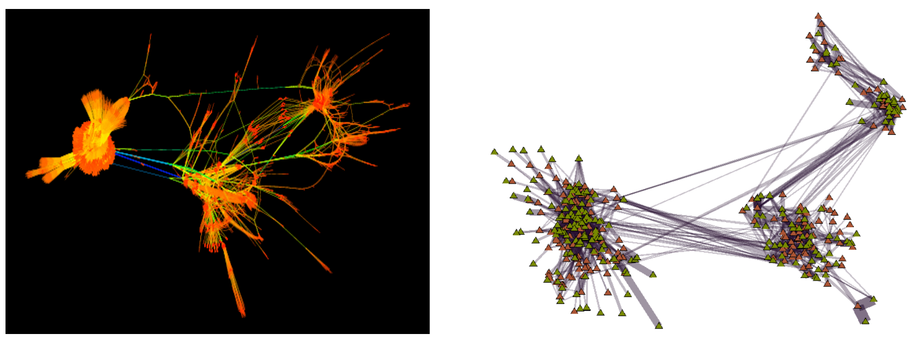
1. Abello, J., Van Ham, F., Krishnan, N.: Ask-graphview: A large scale graph visualization system. IEEE Transactions on Visualization and Computer Graphics **12**(5), 669–676 (2006)
2. AlQuraishi, M., Koytiger, G., Jenney, A., MacBeath, G., Sorger, P.K.: A multiscale statistical mechanical framework integrates biophysical and genomic data to assemble cancer networks. Nature genetics **46**(12), 1363 (2014)
3. Archambault, D., Munzner, T., Auber, D.: Topolayout: Multilevel graph layout by topological features. IEEE Transactions on Visualization and Computer Graphics **13**(2), 305–317 (2007)
4. Archambault, D., Munzner, T., Auber, D.: Grouseflocks: Steerable exploration of graph hierarchy space. IEEE Transactions on Visualization and Computer Graphics **14**(4), 900–913 (2008)
5. Arleo, A., Didimo, W., Liotta, G., Montecchiani, F.: A distributed multilevel force-directed algorithm. In: International Symposium on Graph Drawing and Network Visualization. pp. 3–17. Springer (2016)
6. Barabási, A.L., Gulbahce, N., Loscalzo, J.: Network medicine: a network-based approach to human disease. Nature reviews genetics **12**(1), 56 (2011)
7. Bartel, G., Gutwenger, C., Klein, K., Mutzel, P.: An experimental evaluation of multilevel layout methods. In: International Symposium on Graph Drawing. pp. 80–91. Springer (2010)

8. Batagelj, V., Brandenburg, F.J., Didimo, W., Liotta, G., Palladino, P., Patrignani, M.: Visual analysis of large graphs using (x, y)-clustering and hybrid visualizations. IEEE Transactions on Visualization and Computer Graphics **17**(11), 1587–1598 (2010)
9. Blum, C., Roli, A., Sampels, M.: Hybrid metaheuristics: an emerging approach to optimization, vol. 114. Springer (2008)
10. Cintra, D., Valejo, A., Lopes, A., Oliveira, M.: Visualization to assist interpretation of the multilevel paradigm in bipartite graphs. In: Submitted Graph Drawing and Network Visualization (2019)
11. Dias, M.D., Mansour, M.R., Dias, F., Petronetto, F., Silva, C.T., Nonato, L.G.: A hierarchical network simplification via non-negative matrix factorization. In: 30th SIBGRAPI Conference on Graphics, Patterns and Images (SIBGRAPI). pp. 119–126. IEEE (2017)
12. Frishman, Y., Tal, A.: Multi-level graph layout on the gpu. IEEE Transactions on Visualization and Computer Graphics **13**(6), 1310–1319 (2007)
13. Grujić, J.: Movies recommendation networks as bipartite graphs. In: International Conference on Computational Science. pp. 576–583. Springer (2008)
14. Guillaume, J.L., Latapy, M.: Bipartite structure of all complex networks. Information Processing Letters **90**(5), 215–221 (2004)
15. Guillaume, J.L., Latapy, M.: Bipartite graphs as models of complex networks. Physica A: Statistical Mechanics and its Applications **371**(2), 795–813 (2006)
16. Hachul, S., Jünger, M.: Drawing large graphs with a potential-field-based multilevel algorithm. In: International Symposium on Graph Drawing. pp. 285–295. Springer (2004)
17. Harel, D., Koren, Y.: A fast multi-scale method for drawing large graphs. In: International Symposium on Graph Drawing. pp. 183–196. Springer (2000)
18. Hinge, A., Richer, G., Auber, D.: Mugdad: Multilevel graph drawing algorithm in a distributed architecture. In: Conference on Computer Graphics, Visualization and Computer Vision. p. 189 (2017)
19. Hu, Y.: Efficient, high-quality force-directed graph drawing. Mathematica Journal **10**(1), 37–71 (2005)
20. Hwang, T., Sicotte, H., Tian, Z., Wu, B., Kocher, J.P., Wigle, D.A., Kumar, V., Kuang, R.: Robust and efficient identification of biomarkers by classifying features on graphs. Bioinformatics **24**(18), 2023–2029 (2008)
21. Newman, M.E.: Scientific collaboration networks. i. network construction and fundamental results. Physical review E **64**(1), 016131 (2001)
22. Noack, A., Rotta, R.: Multi-level algorithms for modularity clustering. In: International Symposium on Experimental Algorithms. pp. 257–268. Springer (2009)
23. de Paulo Faleiros, T., Rossi, R.G., de Andrade Lopes, A.: Optimizing the class information divergence for transductive classification of texts using propagation in bipartite graphs. Pattern Recognition Letters **87**, 127–138 (2017)
24. Pawson, T., Linding, R.: Network medicine. FEBS letters **582**(8), 1266–1270 (2008)
25. Perrot, A., Auber, D.: Cornac: Tackling huge graph visualization with big data infrastructure. IEEE Transactions on Big Data (2018)
26. Pope, A.S., Tauritz, D.R., Kent, A.D.: Evolving multi-level graph partitioning algorithms. In: Proceedings of the IEEE Symposium Series on Computational Intelligence (SSCI) (2017)
27. Sharan, R., Ulitsky, I., Shamir, R.: Network-based prediction of protein function. Molecular systems biology **3**(1), 88 (2007)

28. Shneiderman, B.: The eyes have it: A task by data type taxonomy for information visualizations. In: VL'96: Proceedings IEEE Symposium on Visual Languages. pp. 336–343. IEEE CS Press (1996)
29. Staudt, C.L., Sazonovs, A., Meyerhenke, H.: Networkit: A tool suite for large-scale complex network analysis. Network Science **4**(4), 508–530 (2016)
30. Tang, J., Qu, M., Wang, M., Zhang, M., Yan, J., Mei, Q.: Line: Large-scale information network embedding. In: Proceedings 24^{th} International Conference on World Wide Web. pp. 1067–1077. International World Wide Web Conferences Steering Committee (2015)
31. Toosi, F.G., Nikolov, N.S.: Vertex-neighboring multilevel force-directed graph drawing. In: IEEE International Conference on Systems, Man, and Cybernetics (SMC). pp. 002996–003001. IEEE (2016)
32. Valejo, A., Oliveira, M.C.F., Geraldo Filho, P., Andrade Lopes, A.: Multilevel approach for combinatorial optimization in bipartite network. Knowledge-Based Systems **151**, 45–61 (2018)
33. Valejo, A., Valverde-Rebaza, J., de Andrade Lopes, A.: A multilevel approach for overlapping community detection. In: Brazilian Conference on Intelligent Systems. pp. 390–395. IEEE (2014)
34. Walshaw, C.: A multilevel algorithm for force-directed graph drawing. In: International Symposium on Graph Drawing. pp. 171–182. Springer (2000)
35. Wong, P.C., Mackey, P., Cook, K.A., Rohrer, R.M., Foote, H., Whiting, M.A.: A multi-level middle-out cross-zooming approach for large graph analytics. In: 2009 IEEE Symposium on Visual Analytics Science and Technology. pp. 147–154. IEEE (2009)

Appendix A: A case study

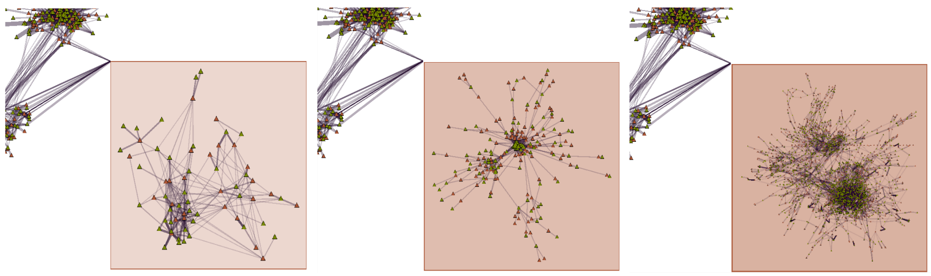
The interaction patterns between genes and proteins cast the basis of molecular biology and of disease pathogenesis [2, 24]. Protein groups are defined by topological characteristics yield by the networks they entail, allowing for the isolation of functional and disease pathways [6, 27]. Given these premises, a brief case study was performed considering the protein and n-reactome iterations network, which has 8,788 proteins, 15,433 iterations, and 41,087 links. Figure 6(a) depicts the original network. One observes three more expressive communities and several smaller communities. Figure 6(b) illustrates a coarsened representation with 100 nodes in each layer obtained after three contraction steps. The coarsened network emphasizes the larger communities, mirroring predominant characteristics of the input network.



(a) original (b) coarsened

Fig. 6: Graphical representations of a network with 24,211 nodes (8,788 proteins and 15,422 n-reactome iterations) and 41,087 links. (a) shows the original network; (b) shows the coarsest network represented with 200 nodes only (100 nodes in each layer).

For this case study, we focused on visualizing the smallest community in order to observe its connectivity and functional node patterns. Figures 7(a), (b) and (c) depict the community observed in levels 2, 1 and 0 (original), respectively. The final result (Figure 7(c)) is displayed in Figure 8, where a group of densely connected nodes can be observed in the central area. An analyst may wish, e.g., to target the genes among such nodes in further steps of the investigation, due to their central role in interacting with numerous proteins. Another possibility is to examine the less connected nodes in an attempt to understand their specific roles. The example illustrates the potential usefulness of BiNetVis in studying pathogenic variants, groups of genes associated with metabolic deficiency, or groups of proteins that can be targeted for therapeutic or other purposes.



(a) level 2 (b) level 1 (c) level 0

Fig. 7: Progressive visualization of the smallest community in Figure 6. In (a) the community is shown at the coarsest level; in (b) it has been been uncoarsened to the next level; and in (c) it has been uncoarsened to the final level, as in the original network.

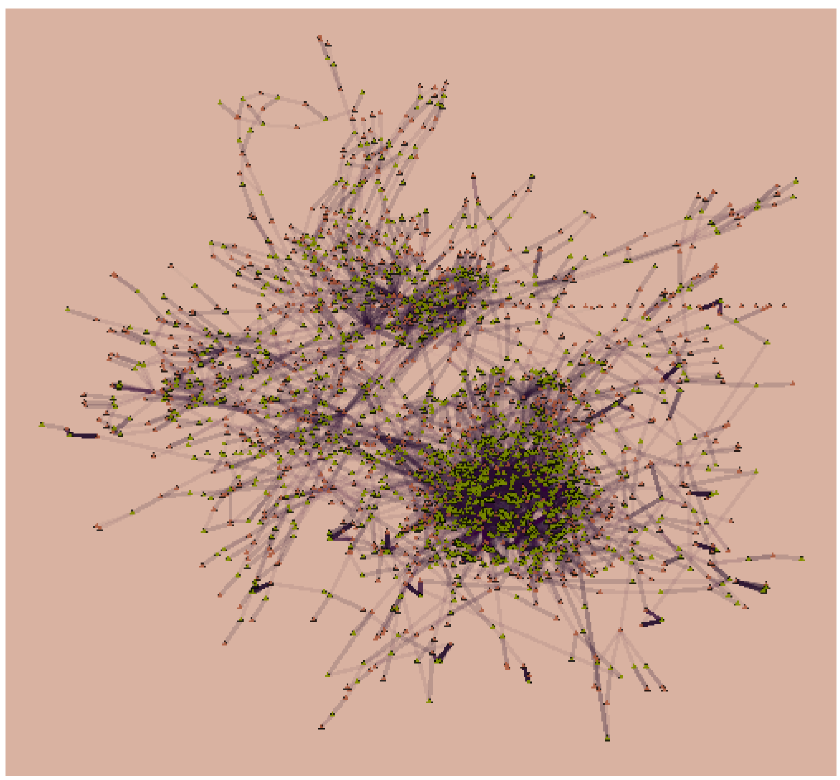


Fig. 8: An isolated visualization of the community under analysis. The researcher reached this visualization by on-demand transformations of the initial visualization of the original network (24,211 nodes) coarsened to a representation with only 200 nodes.

Appendix B: The toolbar

The interface element that enables the interactive visualization in BiNetVis is the toolbar. It was contextualized in Figures 3 and 5 and in Section 3.1, but in a scale which may not favor its inspection by the reader, specially in a printed version of this document. Figure 9 details the toolbar.

Fig. 9: The toolbar, contextualized in Figures 3 and 5, shown independently. The four groups of buttons are for specifying transformations applied to nodes, links, special/multilevel, and whole network. The usage has been described in Section 3.1.

Appendix C: Coarsening parameters

Table 1 lists the input parameters of the coarsening algorithms, which can be applied independently to one or to both network layers. Three parameters are common to all algorithms, namely the maximum number of coarsening levels *max-levels*, a reduction factor *reduction* and a similarity function *similarity*, which returns a similarity score between a pair of vertices. Parameter *reduction* is multiplied by the number of vertices to determine the maximum number of nodes matched, e.g. if $reduction = 0.5$ each coarsening iteration will (potentially) reduce the number of nodes by a factor of two, yielding a logarithmic decrease in network size along the process. A possible similarity function is the *common neighbors* metric, which returns the number of common neighbors between the two nodes. The default coarsening algorithm in the system is $MLPb$, which requires four additional parameters. Parameter *min-vertices* informs the minimum number nodes expected in the coarsest network. Parameters *max-levels* and *tolerance* define stopping criteria: the algorithm stops if either the maximum number of iterations has been reached or the number of label updates is bellow the tolerance threshold. A final parameter, namely *upper bound*, defines an upper limit on the weights of super-nodes, which is useful to preserve the original structures when creating super-nodes and to avoid super-nodes with weights that deviate significantly from the average.

Table 1: Parameters of the multilevel coarsening algorithms (see Section 2.1). The rightmost column reports the default values.

Parameter	**Domain**	**Default**
similarity	string	Common Neighbor
reduction	$(0, 1] \subset \mathbb{R}_+$	0.1
max-levels	$[0, n] \subset \mathbb{Z}_+$	5
min-vertices	$[0, n-1] \subset \mathbb{Z}_+$	100
upper-bound	$(0, n] \subset \mathbb{R}_+$	0.1
iterations	$\subset \mathbb{Z}_+$	10
tolerance	$\subset \mathbb{R}_+$	0.0001