BioNetLink - An Architecture for Working with Network Data

Matthias Klapperstück[1,*] and Falk Schreiber[1,2]

[1]IPK Gatersleben, Corrensstr. 3, OT Gatersleben, 06466 Stadt Seeland, Germany

[2]Martin Luther University Halle-Wittenberg, Von-Seckendorff-Platz 1, 06120 Halle, Germany

Summary

The visualization of biological data gained increasing importance in the last years. There is a large number of methods and software tools available that visualize biological data including the combination of measured experimental data and biological networks. With growing size of networks their handling and exploration becomes a challenging task for the user. In addition, scientists also have an interest in not just investigating a single kind of network, but on the combination of different types of networks, such as metabolic, gene regulatory and protein interaction networks. Therefore, fast access, abstract and dynamic views, and intuitive exploratory methods should be provided to search and extract information from the networks.
This paper will introduce a conceptual framework for handling and combining multiple network sources that enables abstract viewing and exploration of large data sets including additional experimental data. It will introduce a three-tier structure that links network data to multiple network views, discuss a proof of concept implementation, and shows a specific visualization method for combining metabolic and gene regulatory networks in an example.

1 Introduction

Nowadays the work with biological networks plays an essential role in the area of systems biology. Networks provide an insight into the structure of biological systems, can be used to simulate their dynamics, and have also proven useful as basis for visualizing experimental data in their topographical context. Different kinds of networks exist to cover specific biological areas such as metabolism, gene regulation, protein interaction, and so on. Each of the related networks is predestined to show certain kinds of experimental data. For example, metabolic measurements are best shown by using a metabolic network. As another example, to visualize gene expression data one can either use the metabolic network (for genes coding enzymes) or a gene regulatory network, depending on the questions asked. The metabolic network answers questions related to reactions containing the genes such as where in the pathways the gene is showing up or which part may not be reachable anymore once the gene (enzyme) is not present. The gene regulatory network answers questions of how the genes are regulated by transcription factors and show network properties such as connectivity, distance, hubs, etc., that can be further analyzed.

[*]To whom correspondence should be addressed. Email: klapper@ipk-gatersleben.de

The trend for mapping experimental data onto one network at a time is shifting towards a combined view, by integrating multiple networks together to gain a broader view and show the data in an integrated context [1, 2, 3]. As for the example above, using gene expression data usually metabolic networks and gene regulatory networks can be put in such an integrated context, showing not only which reaction is influenced by a gene but at the same time showing what transcription factors may influence this reaction indirectly.

In common software tools usually networks are loaded and held in background using typical network data structures. The information is then used to create mostly static views of the networks. The results are optimal when working on such views for further analysis and exploration [4, 5]. But with the given sizes of currently available networks the ability to navigate and explore them decreases. For example, the metabolic network derived from AraCyc [6] and the gene regulatory network derived from AGRIS [7] contain information about several thousand elements and interactions.

To circumvent such problems arising with large integrated networks it becomes necessary to support dynamic changes of the view presented to the user in near real time. For example build and show abstract groups of elements in the network when a user needs an overview about the network structure, or create views showing filtered elements with a certain focus to them, for example mapped experimental data. To perform such dynamic tasks, data structures have to be present which support the extraction of information and networks with low latency including the creation of suitable layouts. Taking these requirements into account two areas need greater attention: the network data model and a network dependent layouter. An additional area comes into play to support keeping detailed information about each integrated network as well as the information that linked them together.

Graph (or network) based programs usually use standard graph data structures such as adjacency matrices or adjacency lists. These are optimal for working on graph structures and performing typical network analysis tasks. But for structured extraction of specific information or groups of elements for an interactive interplay between analysis and visualization, the structure of these data models can be improved. For example, to extract relevant parts out of a metabolic network one can either scan the whole graph data or use the means of suitable lookup mechanisms. In the latter case, hierarchical access structures can give an abstract and optimized way to extract the wanted information.

Here we introduce a conceptual framework for handling and combining multiple network sources that enables abstract viewing and exploration of large data sets including additional experimental data. We will introduce a three-tier structure that links network data to multiple network views. An example implementation of the concept was written as an Add-on for the VANTED platform [5, 8].

2 Concept

The concept is based on the three-tier architectural model, where the system is separated into three distinct layers (see Fig. 1). They consist of (i) a backend tier, comprising a data reader and a data model, (ii) a logic tier, responsible for data handling and extraction, and (iii) a visualization tier, which creates graphical representations of the data, provided by the second layer. The approach of this separations of the data model and the graph views aims at an increase on access speed for loading, extracting, and visualizing networks.

The concept is designed for reusability. The backend and logic tier can be used independently, since they only read and extract data. Only the visualization tier must be partially adapted for an integration into a new software system, to support specifics in creating views and layouts.

The following sections will explain each part of the three-tier model seen in Fig. 1. The model is intended to be implemented for each network source to reflect the specifics about each network regarding the data structure and visualization.

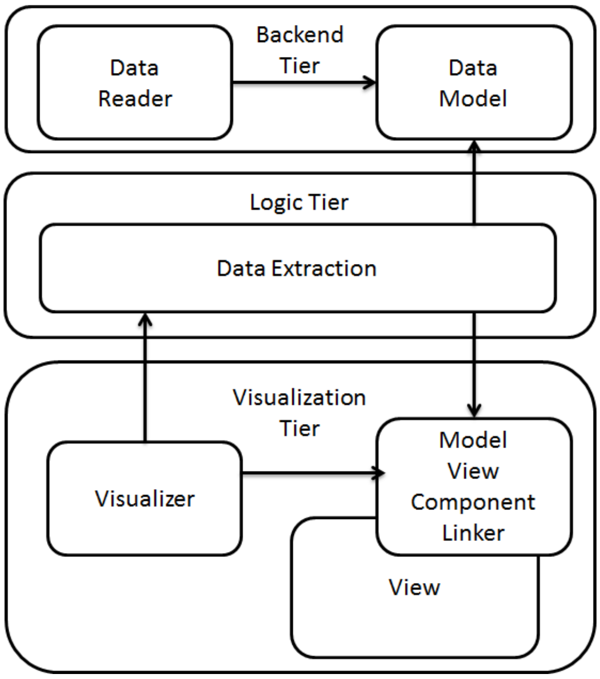

Figure 1: The three-tier structure for reading, extracting, and visualizing network data models.

2.1 The Backend Tier

The backend consists of a data reader and a data model component. The data reader will be specifically implemented for each source available. Such a data source could be, for example, a downloadable data dump file, files using standards such as SBML [9], BioPax [10], or GML [11], or online resources with a query interface.

The data sources will be read and loaded into a network specific data model. This model not only stores the data, but also provides an interface for basic data access. It reflects the inherent structure of the specific network, such as single elements, groups, and connections, which themselves can create hierarchies.

For example, for a metabolic network *single elements* are metabolites and enzymes, *groups* are reactions which combine multiple single elements as well as the connections between them.

Groups of single reactions then create a *pathway*. Going further up in the hierarchy results in *super-pathways*, which group distinct pathways together.

The model reflects such hierarchies by abstracting the elements, building a recursive father-child relationship, providing access to different levels including detailed access to the contained data.

2.2 The Logic Tier

Access to the backend data (located in the backend tier) is provided by using a data extractor, which is contained in the logic tier. It provides high level access to the data taken from the backend with regard to the data context. It therefore has knowledge about the structure of the backend data.

Taking a metabolic backend model as an example, the extraction algorithm would take a list of elements to be selected in the model and then extract a context specific neighborhood. The context specific neighborhood depends on the requested extraction level, which corresponds to the available levels in the data model (see Sect. 2.1). For example, given a single element such as a reaction, the algorithm would extract elements that comprises all the associated metabolites and reactions which are in the pathway the reaction belongs to.

Taking a regulatory network (such as a gene regulatory network) as another example, an extraction algorithm would take a list of elements and extract a context specific neighborhood, which in this case would consist of regulating or regulated elements. The result of such a query would not only contain the sole elements but also information about their interaction (such as the type of regulation), since this is a key feature of such a network.

This logic tier will primarily be used by the visualization tier for creating a context dependent view of the network. For specific tasks and especially new network models in the backend tier adapted or novel extraction algorithms may be necessary and have to be implemented in the logic tier.

2.3 The Visualization Tier

The visualization tier is responsible for creating the actual graphical representation by utilizing a network specific visualizer as well as to set up a model view component linker (MVCL) (see Fig. 1). This MVCL object is a lookup directory, which provides a bidirectional link (and thereby a direct, fast access) from the view to the data model elements and vice versa. Each separate view presented to the user has its own linker, which links the embedded elements back to the same underlying model.

Using the MVCL object, multiple views based on the same data can be created without the need for copying the backend data each time and also provide the independence of the architecture of the graph drawing application. To integrate a previous implementation into a new platform, only the MVCL has to be adapted to support the software specific views.

The visualizers can be separated into two classes, comprising of (i) intra-network and (ii) inter-network visualizers (see Fig. 2) which are described in the following paragraphs.

2.3.1 Intra-network visualization

Each network model will have its own visualizer. Depending on the underlying model, as well as additional information, it will represent the network with a suitable layout. This could be computed by a specific layout algorithm (for a brief overview see [12], or for a generic approach see [13]), or be given by existing layout information (for example, in KGML [14, 15] or SBGN-ML [16] files).

2.3.2 Inter-network visualization

Certain elements appear in different networks. For example, gene identifiers may be used in gene regulatory networks (describing genes) and metabolic networks (describing enzymes coding genes). A combination of these networks will result in a broader context for each element found in the networks.

To visualize the combined network, a separate visualizer performs this task (see Fig. 2). It can utilize the provided intra-network visualizers, the extraction algorithms from the logic tier, or a combination of both to create a suitable layout. An inter-network visualizer needs to know what kind of networks it is merging. As such, it is implemented for a known set of network types. This allows to use background knowledge of each separate network and layout them in a visual appealing and useful way. The inter-network visualizer will not only create a separate model view component linker (MVCL), but also a network interlink object, which keeps track of information about the linked elements of each network type. This information can later be used for fast back-link to the separate network data model.

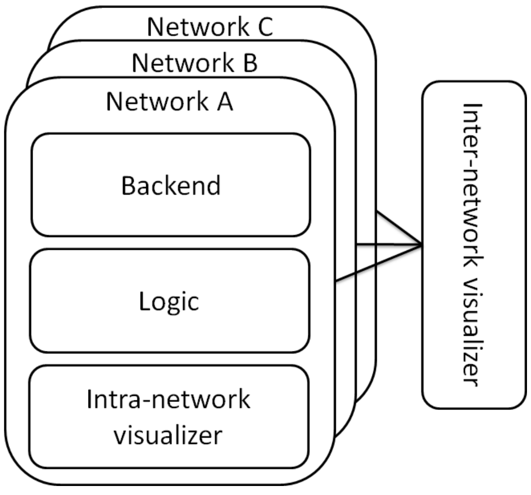

Figure 2: The intra-network and inter-network visualizer in the model concept.

Both, the visualization and the logic tier, build powerful components for selective representation of (parts of) large networks. The selection process can be used in automated tasks, such as selecting elements with embedded experimental data and presenting this data in an inter-network context, or manual triggered tasks, such as a user interaction for displaying a specific subset and presenting it with a new visualization.

3 Implementation

A proof of concept implementation was done to investigate the usability of the conceptual framework. We used plant related information with a focus of transcriptomics data analysis for Arabidopsis. The networks used were derived from two publicly available data sources: the AGRIS gene regulatory network [7] and the AraCyc metabolic network [6]. The data sources consisted of downloadable data dumps from the websites. For both data sources, all three tiers (backend, logic, and visualization tier) as well as an inter-network visualizer combining both networks were implemented.

3.1 Three-tier model for the metabolic network

The AraCyc network reflects the typical hierarchy in a metabolic network, presented as *enzymes* and *metabolites*, *reactions*, *pathways*, and *super pathways*. The data dump also contained layout position information for the metabolites. The same hierarchy is reflected by the implemented model. Groups of elements of the same level are stored in maps, where each element is identified by a unique id, and lists, containing the same elements for iterative purposes. The usage of more than one data structure provides time-optimal access for given tasks. These tasks cover access to single elements in nearly constant time using hash maps, such as retrieving a single reaction by a given id, or groups of elements such as all reactions that belong to a pathway.

The data extraction in the logic tier is able to collect information on different levels (e. g., metabolites, reactions, pathways) from the model. It can select single elements or whole pathways. It strongly uses the provided data structures, implemented in the data model.

AraCyc pathways are shown as separate identities. To be able to connect pathways, the extractor is also able to link pathways based on information found in the model (such as common metabolites). They can then be viewed in a linked context with their adjacent pathways.

The visualizer in the visualization tier is creating the network view, using the position information of the metabolites for the layout. The positions of reactions are derived by calculating the centroid of participating metabolites of that reaction, and positioning the actual reaction element on that centroid. An example is shown in Fig. 3. The visualizer also sets up the lookup directory (MVCL). This lookup directory is using bidirectional maps to link the model entries to views and vice versa. Each view element will have a link to the correnspondent element in the data model. With that link it is for example possible to select a view element, representing a reaction, and retrieve from the model not only the corresponding pathway, but also all containing reactions of that pathway. By using hash maps, these actions can be done in near constant time, depending on the size of the map and its implementation. As a result, the application of standard graph traversal algorithms such as BFS or DFS for retrieving the neighbourhood of interest before the model data is extracted, can be circumvented.

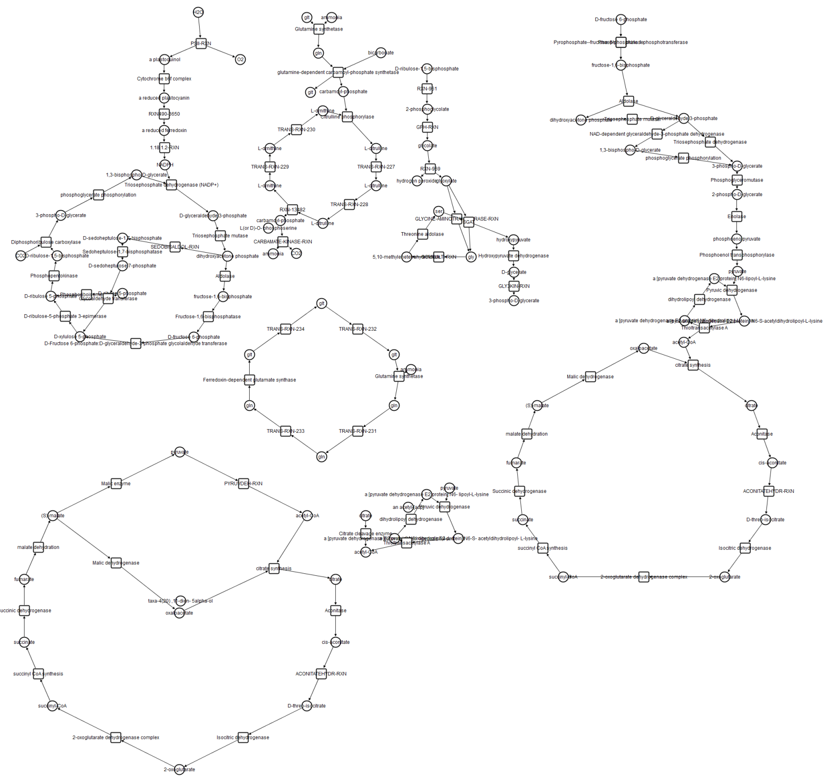

Figure 3: An example visualization of the AraCyc data. Note that although incomplete layout information is given the visualizer of the visualization tier computes complete layouts of the network.

Furthermore additional meta data, such as links between pathways representing neighbours, successors and predecessors, contained classes and so on, can be accessed and individually visualized without having all information and connections represented in one view as one graph.

3.2 Three-tier model for the regulatory network

The backend tier will contain the model of the AGRIS gene regulatory network, with the focus on two distinct element classes and their specific regulation. The two classes are *genes* and *transcription factors*. To reflect the type of regulation, the model stores information about each interaction. The data model stores each gene and transcription factor in a seperate hash map. A third map is created, which contains a TF as key and a list of genes this TF is regulating as a value.

The data extraction in the logic tier is able to extract a neighborhood of selected elements. This neighborhood depends on the type of the selection: if a gene is selected it will get all transcription factors (TF) influencing this gene. If a TF is selected it will extract its influencing TF, but optionally also all genes which are influenced by this TF. As the neighborhood of more than one element can be extracted the result of the extraction algorithm may be number of (small) unconnected neighborhoods. In this case the algorithm tries to find regulations between elements of these neighborhoods to create one single regulatory network.

The visualization tier will take the extracted data elements and sets up the MVCL to create the bidirectional link between the graph elements and the data model. It alsod performs a force-directed layout [17] on the constructed network view as well as setting the shapes and colors of regulatory edges according to the visual styles used in the AGRIS online interface [18]. An example is shown in the Fig. 5, here the central part is a subnetwork of the AGRIS network laid out based on a force-directed layout.

3.3 Inter-network visualizer

To view mapped data in a combined context of more than a single network, a separate inter-network visualizer was implemented, as mentioned in Sect. 2.3.2. It will use the specific layouts of each network but also use combinatory layout of the different networks in context of one view.

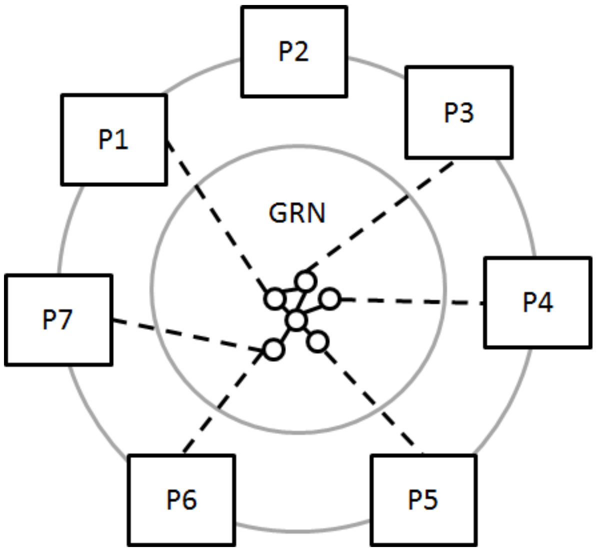

Figure 4: The result of the inter-network visualizer with the gene regulatory network in the center surrounded on a circle by pathways. The dotted edges show the connections between elements of both networks.

As for the given regulatory and metabolic network, a combination of two layout algorithm was applied, see Fig. 4. The regulatory network is placed in the center and layouted using the force-directed method. For the metabolic network first all relevant pathways (for example pathways containing mapped data) and their interconnections are computed. For each pathway a bounding box is calculated and the extracted pathways are placed on a circle around the central network. To highlight links between elements of the two networks, interlink edges between matching targets are created.

The visualizer will also create a MVCL, binding each graph element to the related model, in this case part AraCyc and part AGRIS. This linker will be used, during visual graph exploration tasks, where the user will be able to select elements of interest, and create new views with a different focus and or layout, without changing the data model.

4 Application and results

As application example we show transcriptomics data in the context of combined metabolic and gene regulatory networks. The transcriptomics data has been pre-analyzed to express a specific focus. The result is a list of interesting genes which should be further visually analyzed. Many algorithms can be used to produce lists of relevant genes such as highly expressed genes, strongly correlated genes and so on. In our case all genes which show a high correlation to a specific metabolite over different experimental conditions have been selected. The result can be seen in Fig. 5.

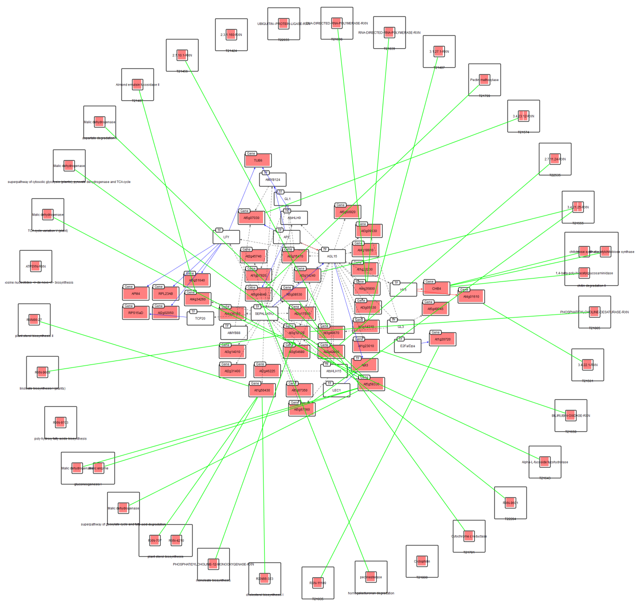

Figure 5: The result for mapped data (selected genes) on the gene regulatory network (in the center) and the surrounding abstract metabolic network. The edges between the networks indicate the same data element (gene Id) found in both networks.

The figure shows an application of the integrated layout concept mentioned in the previous chapter. In the center the gene regulatory network, consisting of genes, transcription factors, and their interaction, is shown. It consists of the selected genes of interest and their direct neighborhood. This network is surrounded by an abstracted metabolic network view, where only the reaction nodes are shown. These reaction nodes are surrounded by an artificial node that represent the relevant pathway. The reactions inside each artificial pathway node are laid out in a quadratic grid. The edges that connect nodes between the inner regulatory network and the outer circular metabolic network link elements containing the same gene id. Nodes, containing mapped data are drawn in red, showing a positive correlation value to a specific metabolite as a heatmap.

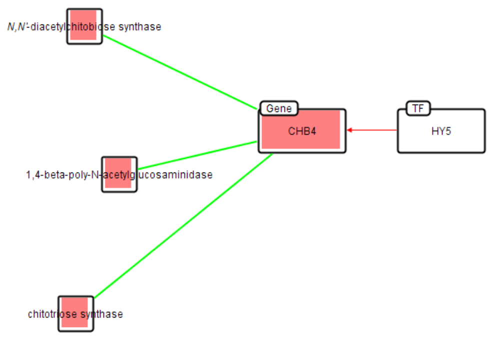
N,N'-diacetylchitobiose synthase
Gene
CHB4
TF
HY5
1,4-beta-poly-N-acetylglucosaminidase
chitotriose synthase

Figure 6: An extracted view, created using a user selection of an enzyme in the network with its corresponding reaction shown in Fig. 5.

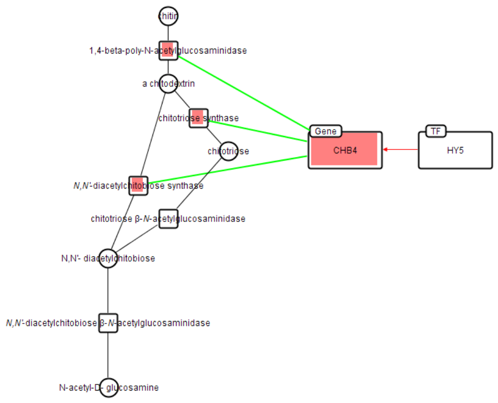
chitin
1,4-beta-poly-N-acetylglucosaminidase
a chitodextrin
chitotriose synthase
chitotriose
Gene
CHB4
TF
HY5
N,N'-diacetylchitobiose synthase
chitotriose β-N-acetylglucosaminidase
N,N'- diacetylchitobiose
N,N'-diacetylchitobiose β-N-acetylglucosaminidase
N-acetyl-D- glucosamine

Figure 7: An alternative extracted view, showing the full pathway, created using a user selection of an enzyme in the network shown in Fig. 5.

The visualization shown in Fig. 5 shows all mapped data in context of the networks. If the user is particularly interested in one reaction or gene, and its inter-network context, the user can select that element and another inter-network visualizer is used again on the current selection. The resulting view a subset of the original network, that focusses on one pathway with its containing reactions and the gene regulation. This is shown in Fig. 6. Another version of this visualization includes the whole pathway in combination with the gene regulation. The visualizer thereby uses the layout found in the metabolic data model and does a new force directed layout on the extracted regulatory network. In VANTED the result of the subset networks is shown in a separate view.

We performed basic measurements of the runtime for specific tasks such as creation of combined network and subset network views we could see, that the data retrival using the the second tier, didn't use significant computation time (less than 5% depending on the number of elements to lookup). Most of the time was used by the creation of the software specific graph view components as well as the layout algorithms. It can be expected, that there is no significant increase in computation time, when using standard graph search algorithms, since the network size is small enough, regarding metabolic or regulatory networks. But using the structured models and specific data extraction methods helped significantly to implement new intra and inter-network visualizers during work progress.

5 Future work and conclusion

Currently the visualization is bottom up, i. e., based on the network data model and experimental data the networks are created and laid out on the fly and presented to the user. If the user decides to edit, delete, or insert certain elements, then these changes are not reflected back to the model. A listener concept has to be implemented to follow user actions that will be transferred back to the model (to keep view and model in sync).

The design of the data model which includes hierarchies is the basis for adaptive zooming [19], which would enable the user to browse networks with a huge number of elements, such as the AraCyc network. Depending on the zoom level the user would either see only pathway nodes (for an overview) and when zooming in, single reactions would become visible. Adaptive zooming becomes a challenging visualization when working with networks and mapped data as a proper visual non-distorting representation has to be found.

The concept of a three-tier model, as presented in this paper, has been proven useful in combining and layouting different types of networks. By using the MVCL lookup tables, besides a fast bidirectional model-view access, a clean separation of model data and visualization was achived enabling future developers to apply and adapt the three-tier model for their analysis platforms. Using the structured data models, which reflect the network topology is helpful for optimal data extraction and view creation tasks, such as in creating full or subset network views.

References

[1] A. Gutteridge, M. Kanehisa and S. Goto. Regulation of metabolic networks by small molecule metabolites. *BMC Bioinformatics*, 8(1):88, 2007.

[2] T. Y. Kim, H. U. Kim and S. Y. Lee. Data integration and analysis of biological networks. *Current Opinion in Biotechnology*, 21(1):78–84, 2010.

[3] E. Watson, L. MacNeil, H. Arda, L. Zhu and A. Walhout. Integration of metabolic and gene regulatory networks modulates the C.elegans dietary response. *Cell*, 153(1):253–266, 2013.

[4] C. Huttenhower, S. Mehmood and O. Troyanskaya. Graphle: Interactive exploration of large, dense graphs. *BMC Bioinformatics*, 10(1):417, 2009.

[5] H. Rohn, A. Junker, A. Hartmann, E. Grafahrend-Belau, H. Treutler, M. Klapperstück, T. Czauderna, C. Klukas and F. Schreiber. VANTED v2: a framework for systems biology applications. *BMC Systems Biology*, 6(1):139, 2012.

[6] L. Mueller, P. Zhang and S. Y. Rhee. AraCyc: a biochemical pathway database for Arabidopsis. *Plant Physiology*, 132(2):453–460, 2003.

[7] A. Yilmaz, M. Mejia-Guerra, K. Kurz, X. Liang, L. Welch and E. Grotewold. AGRIS: the Arabidopsis Gene Regulatory Information Server, an update. *Nucleic Acids Research*, 39(1):D1118–D1122, 2011.

[8] C. Klukas and F. Schreiber. Integration of -omics data and networks for biomedical research with VANTED. *Journal of Integrative Bioinformatics*, 7(2):112, 2010.

[9] M. Hucka, A. Finney, H. M. Sauro et al. The systems biology markup language (SBML): a medium for representation and exchange of biochemical network models. *Bioinformatics*, 19(4):524–531, 2003.

[10] E. Demir, M. P. Cary, S. Paley et al. The BioPAX community standard for pathway data sharing. *Nature Biotechnology*, 28(9):935–942, 2010.

[11] M. Himsolt. A portable Graph File Format. Technical report, University of Passau, 1996.

[12] A. Kerren and F. Schreiber. Network visualization for integrative bioinformatics. In M. Chen and R. Hofestädt (editors), *Approaches in Integrative Bioinformatics*, pages 173–202. Springer Berlin Heidelberg, 2014.

[13] F. Schreiber, T. Dwyer, K. Marriott and M. Wybrow. A generic algorithm for layout of biological networks. *BMC Bioinformatics*, 10:375, 2009.

[14] C. Klukas and F. Schreiber. Dynamic exploration and editing of KEGG pathway diagrams. *Bioinformatics*, 23(3):344–350, 2007.

[15] M. Kanehisa, S. Goto, Y. Sato, M. Furumichi and M. Tanabe. KEGG for integration and interpretation of large-scale molecular data sets. *Nucleic Acids Research*, 40(1):D109–D114, 2012.

[16] M. P. van Iersel, A. C. Villeger, T. Czauderna et al. Software support for SBGN maps: SBGN-ML and LibSBGN. *Bioinformatics*, 28(15):2016–2021, 2012.

[17] T. Fruchterman and E. Reingold. Graph drawing by force-directed placement. *Software - Practice and Experience*, 21(11):1129–1164, 1991.

[18] GRG-X: Grassius regulatory grid explorer. `http://arabidopsis.med.ohio-state.edu/grgx/`.

[19] A. Cecconi and M. Galanda. Adaptive zooming in web cartography. *Computer Graphics Forum*, 21(4):787–799, 2002.

[20] E. Huala, A. Dickerman, M. Garcia-Hernandez et al. The arabidopsis information resource (TAIR): a comprehensive database and web-based information retrieval, analysis, and visualization system for a model plant. *Nucleic Acids Research*, 29(1):D102–D105, 2001.

[21] F. Schreiber, C. Colmsee, T. Czauderna, E. Grafahrend-Belau, A. Hartmann, A. Junker, B. H. Junker, M. Klapperstück, U. Scholz and S. Weise. MetaCrop 2.0: managing and exploring information about crop plant metabolism. *Nucleic Acids Research*, 40(1):D1173–D1177, 2012.

[22] M. A. Westenberg, S. A. F. T. Hijum, A. T. Lulko, O. P. Kuipers and J. B. T. M. Roerdink. Interactive Visualization of Gene Regulatory Networks with Associated Gene Expression Time Series Data. In L. Linsen, H. Hagen and B. Hamann (editors), *Visualization in Medicine and Life Sciences*, Mathematics and Visualization, pages 293–311. Springer Berlin Heidelberg, 2008.