Network Visualization for Integrative Bioinformatics

Andreas Kerren and Falk Schreiber

Abstract Approaches to investigate biological processes have been of strong interest in the past few years and are the focus of several research areas like systems biology. Biological networks as representations of such processes are crucial for an extensive understanding of living beings. Due to their size and complexity, their growth and continuous change, as well as their compilation from databases on demand, researchers very often request novel network visualization, interaction and exploration techniques. In this chapter, we first provide background information that is needed for the interactive visual analysis of various biological networks. Fields such as (information) visualization, visual analytics and automatic layout of networks are highlighted and illustrated by a number of examples. Then, the state of the art in network visualization for the life sciences is presented together with a discussion of standards for the graphical representation of cellular networks and biological processes.

Keywords biological networks, visualization, graph drawing, visual analytics, interaction, exploration, SBGN, visualization tools

1 Introduction

Many biological processes are represented as networks. Examples are networks from the area of molecular biology such as metabolic networks, protein interac-

Andreas Kerren
Linnaeus University, Department of Computer Science, Vejdes Plats 7, SE-351 95 Växjö, Sweden
e-mail: andreas.kerren@lnu.se

Falk Schreiber
Martin Luther University Halle-Wittenberg, Von-Seckendorff-Platz 1, D-06120 Halle, Germany, and IPK Gatersleben, Corrensstrasse 3, D-06466 Gatersleben, Germany, e-mail: schreibe@ipk-gatersleben.de

In M. Chen and R. Hofestädt (Eds.), Approaches in Integrative Bioinformatics, pages 173-202, Springer, 2014.

tion networks, and gene regulatory networks, but also from other areas of the life sciences such as ecological networks, phylogenetic networks, neuronal networks, chemical structures, infection networks and so on. Network modeling, analysis, and visualization are important steps towards a systems biological understanding of organisms and organism communities. The graphical depiction of such networks supports the understanding of the underlying processes and is essential to make sense of much of the complex biological data that is now being generated.

A picture of a network is called a *network diagram* or a *network map*, see Fig. 1 for a SBGN map of a metabolic pathway. A network diagram representing biological processes consists of a set of elements (called *nodes* or *vertices*) and their connections or interactions (called *edges*). These elements and connections often have a defined appearance and are placed in a specific layout. Due to the size and complexity of such networks, methods for their automatic visualization and interactive exploration are desired.

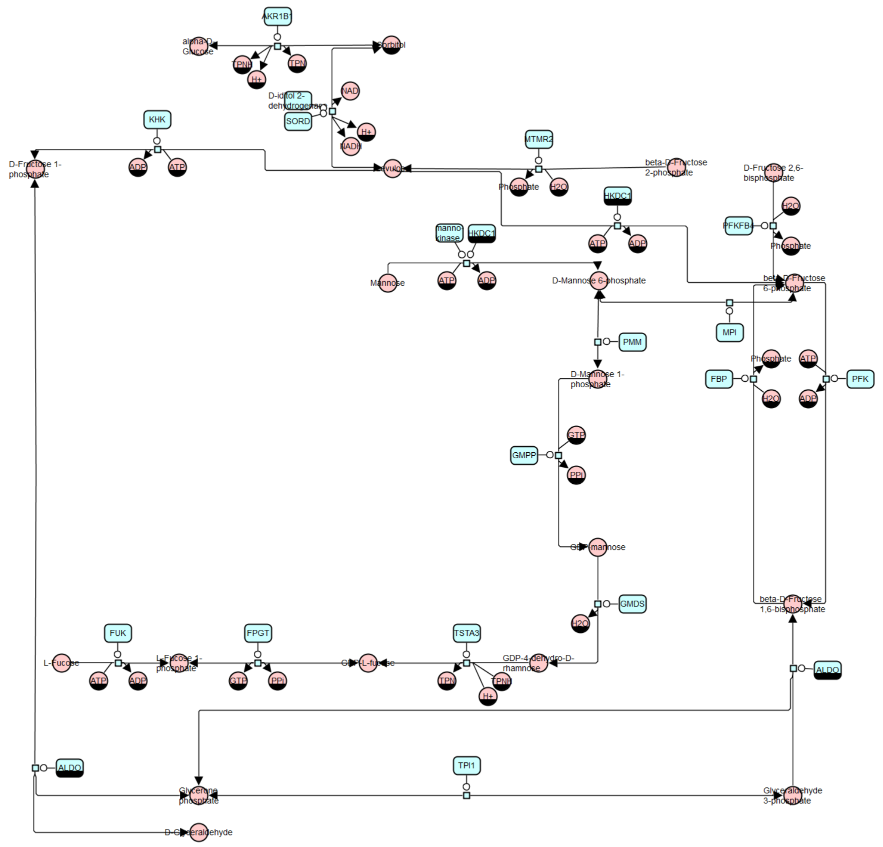


Fig. 1 A map of a metabolic pathway shown in the SBGN standard [87], derived from *KEGG* [61], computed and displayed by *Vanted* [110].

Network diagrams or maps have been produced manually for a long time. Examples are textbooks on biochemistry [8, 96], biological network posters [94, 99], and some electronic information systems such as *ExPASy* [4] and *KEGG* [61]. The drawings in these resources have been created manually long before their use and provide only a restricted view of the data. These maps represent the knowledge at the time of their generation and are static, hence cannot be changed by an end-user. Therefore, this type of biological network visualization is often called *static visualization*.

Because of the size and complexity of biological networks, their steady growth and continuous change, as well as the compilation of user-specific networks from databases, novel automatic visualization, interaction and exploration methods are desired. The generation of a network map on demand is called *dynamic visualization*. Such visualizations are automatically created by the end-user from up-to-date data. Their advantages are, inter alia, that they can be modified to provide particular views at the data, and often navigation and exploration methods are supported in interactive systems.

This review gives a brief introduction into (information) visualization, visual analytics and automatic layout of networks, presents the state of the art in automatic network visualization for the life sciences, and standards for the graphical representation of cellular networks and biological processes. It is structured in two main parts as follows: Section 2 provides information about the foundations from computer science in general and looks into the subareas of *information visualization*, *graph drawing* (network visualization), and *visual analytics* in particular. Section 3 takes a closer look at the visualization of biological networks and discusses methods, some important tools and the SBGN standard. It looks into the application and extension of computer science methods for the special requirements of the life sciences.

2 Background

The effective visualization of biological networks is influenced by research from many different fields. In the past, such networks were simply considered as large graphs (or hypergraphs), and a suitable visual representation was restricted to finding an appropriate (static) graph layout. Nowadays, research in the visualization of large and complex networks is more focused on interactive exploration and analysis that includes the consideration of additional data that might be attached to various graph elements or that might be the basis for the construction of biochemical networks. The process of such a data collection and storage will heavily increase in the future. This is especially true in systems biology where, for example, the huge amount of *omics data automatically generated by high-throughput technologies [3, 39] lead to the challenge of interpreting all of these data sets in context of networks. The fundamental problem today is to transform the data—which is typically not pre-processed, erratic, stored in idiosyncratic formats, sometimes uncertain, and often composed of various types (multidimensional, time-dependent, geo-spatial, ...)—into information and make it useful/available/analyzable to analysts. Often, this challenge is called the *information overload problem*. Positive effects of such a transformation are then to discover something that is interesting (like patterns or outliers) or to monitor a huge data set in real time [73].

Because of this general view on the problem, we provide a more general background section. First, we discuss the field of information visualization in the next subsection. We highlight the most important definitions/aims and present a brief high-level overview of visual representations and interaction techniques. Then, we outline the field of graph drawing and discuss the most often used layout algorithms. Finally, a relatively new field, called visual analytics, is introduced. Due to page limitations, we cannot give a comprehensive overview of all aspects of the aforementioned research fields. Instead, we present a selection of fundamental ideas/approaches and refer to the literature including surveys.

2.1 Information Visualization

Information Visualization (InfoVis) is a research area which focuses on the use of interactive visualization techniques to help people understand and analyze data. While related fields such as scientific visualization involve the presentation of data that has some physical or geometric correspondence, information visualization centers on abstract information without such correspondences, i. e., information that cannot be mapped into the physical world in most cases. Examples of such abstract data are symbolic, tabular, networked, hierarchical, or textual information sources. The ever increasing amount of data generated or made available every day amplifies the urgent need for InfoVis tools. To give the field a firm base, InfoVis combines several aspects of different research areas, such as scientific visualization, human-computer interaction, data mining, information design, cognitive psychology, visual perception, cartography, graph drawing, and computer graphics [74, 75].

2.1.1 The Importance of Human Visual Perception and Visual Metaphors

Human information processing and the human capability of information reception have to be adequately taken into account when developing visualization tools. This should be reflected in an appropriate user interface design, a clean requirement analysis and modeling, and perhaps most important an efficient interaction between the human analyst and the computer. Discussing the different features of our eye, the various process models of human visual perception (incl. preattentive perception and features), or our capabilities of pattern recognition would go beyond the scope of this background section. There are many good textbooks that deal with these topics in context of visualizations: we recommend the books of Ware [141], Kerren *et al.* [75] and Ward *et al.* [140].

Edward Tufte, one of the leaders in the field of visual data exploration, describes in his illustrated textbooks [131, 132, 133] how information can be prepared so that the visual representation depicts both the data and the data context. The use of suitable visual metaphors assists our brain in its endeavor to connect new information received through the visual input channels to existing information stored in short- or long-term memory [71]. Tufte inspired many InfoVis researchers in their ambition to develop novel visual representations for the data sets under consideration (the process of representing a concrete data set by an appropriate visual structure is called "visual mapping") as well as interaction techniques which support a better understanding of the data.

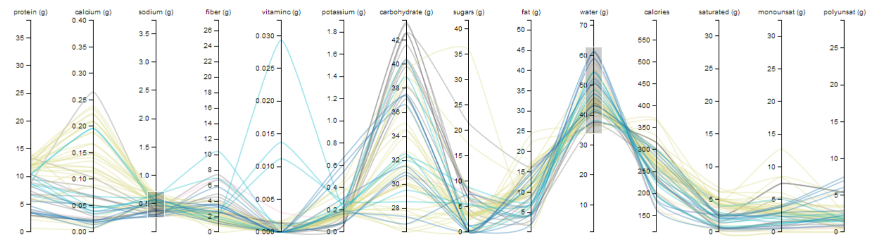
2.1.2 Visual Representations

Visual mappings explain how data models can be expressed using visual metaphors and be converted into corresponding visual representations which are suitable for interaction. This is typically done in the 2D space, because 3D representations usually introduce unnecessary clutter and navigation problems. We highlight the most important visualization techniques for basic data types in the following paragraphs. Of course there are other types of data that have to be considered. We refer to the literature if the reader is interested to get more information, such as [27, 102] for geo-spatial data, [2] for time-series data, or [41, 126, 140] for a comprehensive discussion of visual representations in general.

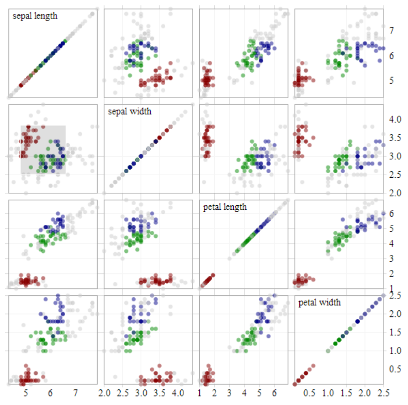
Visualization Techniques for Multivariate Data

Multivariate (or multidimensional) data sets can mostly be described as data tables with *n* data objects and *m* attributes/features, i. e., for each object exists an attribute vector with *m* dimensions. The attribute values can be classified into nominal, ordinal, or quantitative. In practice, we often have a large amount of data objects and many attributes with different types. Finding a suitable visual representation is thus challenging, and the right choice might depend on further parameters like application domain, integration into a larger visualization environment or support of specific interaction techniques. In general, visual mappings for multivariate data can roughly be categorized as follows:

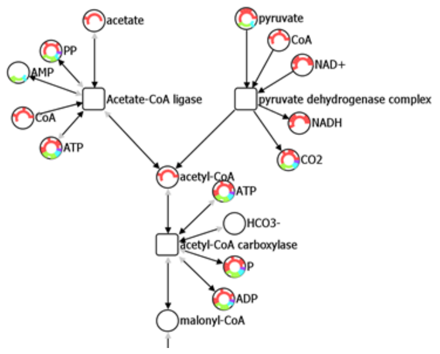
Point-based approaches: This class of techniques projects *n*-dimensional objects from the data space to a lower-dimensional—typically 2D—display space [140]. There are different variations: *scatter plot matrices*, for instance, consist of a grid of 2D scatter plots each showing a possible pair of dimensions/attributes [19], see Fig. 2(b) for an example. Dimensional reduction techniques, such as multi-dimensional scaling (*MDS*) [92, 145], principal component analysis (*PCA*) [53] or self-organizing maps (*SOMs*) [80], project *n*-dimensional data records into 2D/3D directly. The idea is to preserve properties of the multivariate data space



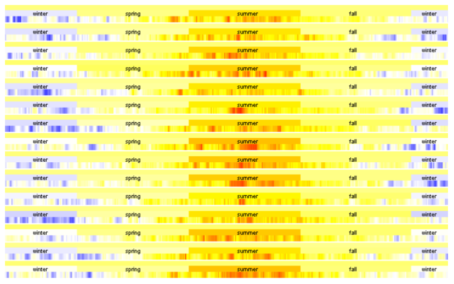
(a) Parallel coordinates that visualize a nutrient content data set with more than 1,000 data objects and 14 attributes (available online [31]). Note that the visible polylines were interactively selected in the 3rd and 10th axis.



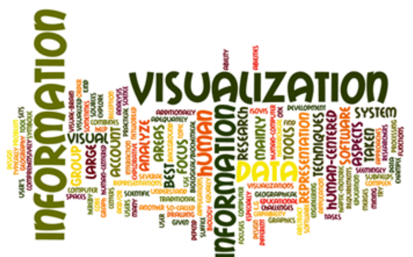
(b) A scatter plot matrix showing data from the Iris data set (available online [11]). Also in this case, the colored points indicate data selected by the user (see the grey colored selection in the plot of the first column, second row).



(c) Small icons/glyphs are embedded into the graph nodes of a metabolic network. In this case, they indicate reachable nodes in other (color-coded) pathways [60].



(d) A pixel-based approach to visualize weather data of a city. The rows represent years, and the temperatures (color-coded from blue over white to red) of each day are ordered from left to right [90].



(e) Sample tag cloud of a text document which is related to information visualization (generated with Wordle [32]).

Fig. 2 Some examples of often used visualization techniques. The screenshots in (a) and (b) were produced with D3 [22].

during the projection, i. e., similar data objects in data space should also be similar in display space which is represented by neighborhood. Note that absolute positions in the display space are less important; in contrast to relative positions.

Axis-based approaches: Here, a multidimensional data object is usually represented by a polyline, and its attribute values are marked on coordinate axes which can be arranged in various ways. Thus, the user can read the attribute values from the intersections between the coordinate axes and the polyline. The most prominent examples are *parallel coordinate systems* [49] (cf. Fig. 2(a)) or *star plots* [16] (also called Kiviat diagrams).

Icon-based approaches: Icon- or glyph-based approaches are coherent graphical entities that represent the attribute values of a data record by modification of the entity's visual features, such as line thickness, size, color, orientation, etc. There are many different realizations, such as *stick figures* [106], *Chernoff faces* [18] or *shape coding* [7]. A variant of so-called *rose diagrams* [100] is shown in Fig. 2(c).

Pixel-based approaches: Such approaches try to maximize the available display space by mapping attribute values to single pixels. There is only one degree of freedom to represent such a value by a pixel: its color. Therefore, the challenge in the development of pixel-based representations is to arrange the used pixels on the screen in a meaningful way. Well-known examples are *recursive patterns* [69] or the *VisDB* tool [67] for the analysis of databases. Fig. 2(d) exemplifies the idea in context of the visualization of weather data collected over time.

Visualization Techniques for Hierarchical Data and Networks

Networks and trees are in the center of our interest in this chapter. Therefore, we provide an own Section 2.2 for a deeper discussion of suitable visualization possibilities for these data types and focus there on traditional node-link approaches. For the sake of completeness, we want to note that there are also so-called space-filling methods that try to solve some conceptual problems of node-link diagrams, such as the high space consumption and difficult inclusion of many (and complex) attributes into the drawing. *Treemaps* fall into this category in which the hierarchy is recursively mapped to rectangular areas [52]. Other examples are *Beamtrees* [134], *sunburst* approaches [108], or *network matrices* [1].

Visualization Techniques for Text and Documents

Today, the availability of texts and documents is overwhelming, and people want to actively deal with them to solve specific problems. Typical questions are: what documents contain a text about a specific topic? Or, are there similar documents to those that I already have? Information visualization is capable of supporting the aforementioned tasks in several ways.

Text visualization: First, we focus on approaches to the visualization of a single text document. *Tag Clouds* provide information about the frequency of words contained in a text [63]. The approach uses different font sizes for each word in the text to indicate how often a certain word is used in comparison with the other words as shown in Fig. 2(e). Several extensions and related approaches exist, such as *Wordle* or *ManiWorlde* [77, 138]. *SparkClouds* extend the original tag cloud idea with a temporal variable by so-called "sparklines" [88]. Thus, trends can easily be identified and analyzed. An approach for visual literary analysis is called *Literature Fingerprinting* [68]. It supports the visual comparison of texts by calculating features (e. g., word/sentence length or measurement of vocabulary richness) for different hierarchy levels and by creating characteristic fingerprints of the texts.

Document visualization: Collections of text documents can be structured to some extent (software packages, wikis, ...) or relatively unstructured (emails, patents, ...). Early approaches, e. g., *Lifestreams* [34], simply arranged documents according to specific attribute values such as time tags. More recent works analyze the documents by metrics, such as similarity, and perform cluster analyses or compute SOMs. Conceptually similar (by looking at the resulting visual representation) is *ThemeScapes* [147] that follows a natural landscape metaphor. Single documents are categorized and then mapped to a document map as topic areas, whereas the documents themselves are shown as small dots. "Mountains" in the landscape represent document concentrations in a thematic environment (density), height lines connect concept domains, etc. There are many more recent approaches that make use of the same metaphor, such as [104]. In order to carry out comparisons of text documents using tag clouds, *Parallel Tag Clouds* [20] arrange tags on vertical lines for each document. Identical words are then highlighted by connection lines.

2.1.3 Interaction Techniques

Interaction techniques in information visualization are mechanisms "for modifying what the users see and how they see it" [140]. There are many taxonomies of interaction techniques in the literature which help to better understand the design space of interaction; a nice overview is provided by [148]. In the following, we present a simplified and shortened classification of interaction methods for information visualization from our paper [73] which is based on [43] of its own.

Data and View Specification This category focuses on the data space and how the data is visually represented (corresponds to data transformations and visual mappings in the InfoVis Reference Model [14]).

- *Encode/Visualize:* Users can choose the visual representation of the data records including graphical features, such as color, shape, etc. Visual representations typically depend on the data types as discussed in Sect. 2.1.2.

- *Reconfigure:* Some interaction techniques allow the user to map specific attributes to graphical entities. An example is the mapping of attributes in a multivariate data set to different axes in a scatter plot.
- *Filter:* This technique is of great importance as it allows the user to interactively reduce the data shown in a view. Popular methods are *dynamic queries* by using range sliders [146] or picking a set of nodes in a network visualization for further analyses by performing a "lasso" selection [44].
- *Sort:* Ordering of records according to their values is a fundamental operation in the visual analysis process. This is, for example, important in network analysis where nodes might be sorted based on specific centrality values [150].

View Manipulation Our second category addresses interacting with visual representations (view transformations in the InfoVis Reference Model).

- *Select:* Selection is often used in advance of a filter operation. The aim is to select an individual object or a set of objects in order to highlight, manipulate, or filter them out. Examples include putting a placemark on a virtual map to highlight a spatial area or the specification of attribute ranges in parallel coordinate systems as seen in Fig. 2(a).
- *Navigate/Explore:* This important class of interaction techniques typically modify the level-of-detail in visualizations following the mantra *overview first, zoom and filter, details on demand* [121]. Well-known approaches are *focus&context* [111], *overview&detail* [51], *zooming&panning* [137], or *semantic zooming* [127].
- *Coordinate/Connect:* Linking a set of views or windows together to enable the user to discover related items. Brushing and linking techniques (e. g., *histogram brushing* [89]) are used in almost all information visualizations, such as in [59].
- *Organize:* Large visualization systems often consist of several windows and workspaces that have to be organized on the screen. Adding and removing views can be confusing to the analyst. Some systems help the user to better overview and to preserve his/her mental map by grouping of views or by assigning specific places where they have to appear [50, 91].

Note that it is possible and also common practice to combine the aforementioned techniques. The given literature references only point to selected example works and make no claim to be complete.

2.2 Graph Drawing and Network Visualization

In this subsection, we distinguish between *graphs* and *multivariate networks*. A (simple) graph $G = (V,E)$ consists of a finite set of vertices (or nodes) V and a set of edges $E \subseteq \{(u,v)|u,v \in V, u \neq v\}$. Whereas, a multivariate network N consists of an underlying graph G plus additional attributes that are attached to the nodes and/or edges. To describe the fundamental ideas of graph visualization algorithms more efficiently, we have to provide some definitions:

- An edge $e = (u, v)$ with $u = v$ is called a *self-loop*.
- If an edge e exists several times in E then it is called a *multiple edge*.
- A *simple graph* has no self-loops and no multiple edges. Here, we assume that all graphs are simple graphs for the sake of convenience.
- The *neighbors* of a node v are its adjacent nodes.
- The *degree* of a node v is the number of its neighbors.
- A *directed graph* (or digraph) is a graph with directed edges, i. e., (u, v) are ordered pairs of nodes.
- A directed graph is called *acyclic* if it has no directed cycles, i. e., there is no directed path where the same node is visited twice.
- A graph is *connected* if there is a path between u and v for each pair (u, v) of nodes.
- A graph is *planar* if it can be drawn in the 2D plane without intersections of edges (*edge crossings*).

2.2.1 Traditional Graph Drawing (GD)

Graph drawing algorithms compute a 2D/3D layout of the nodes and the edges, mainly based on so-called *node-link diagrams* [141]. They play a fundamental role in network visualization. Particular graph layout algorithms can give an insight into the topological structure of a network if properly chosen and implemented. The graph readability is affected by quantitative measurements called *aesthetic criteria* [24], such as

- minimization of edge crossings,
- minimization of the drawing area,
- displaying the symmetries of the graph topology,
- constraining edge lengths,
- constraining the number of edge bends, and
- maximization of the resolution.

Thus, graph drawing generally deals with the ways of drawing graphs according to the set of predefined aesthetic criteria [17]. A problem is that these criteria are often contradictory, and problems which aim to optimize the criteria are often NP-hard. Therefore, many GD algorithms are heuristics. Note that we only focus on traditional GD approaches in this subsection. There are further possibilities to represent graphs, such as matrix representations [1] or hybridizations between both approaches [44] (cf. Sect. 2.1.2).

In the following paragraphs, a selection of drawing approaches is presented. These are layout methods for trees, force-based layout techniques, and hierarchical drawings. There are many more approaches not discussed here, for instance, orthogonal layouts [29], visualization of hypergraphs [9], or dynamic layouts for graphs that change over time [25] (a possible application of dynamic approaches is visualizing the evolution of biochemical networks [112], for instance). Implementing good graph drawing algorithms is usually complicated and time consuming. Therefore, a number of different open source libraries were developed, such as *JUNG* [105] and many others, that allow to simply call predefined methods for the computation of a specific graph layout.

Tree Drawings

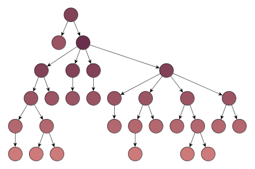
Trees are a special case of directed (acyclic) graphs that usually have a distinguished node called the *root* of the tree. We can regard a tree as a digraph with all edges oriented away from the root. A binary tree is a rooted tree where each node has at most two children (we assume here that binary trees are ordered). The Graph Drawing community developed a lot of different layout methods for binary and general trees. In this context, there is another set of more specified aesthetic criteria especially for (binary) trees:

- Nodes at the same level of the tree should lie along a straight line, and the straight lines defining the levels should be parallel.
- A left subtree should be positioned to the left of its parent node and a right subtree to the right.
- A parent node should be centered over its subtrees.
- Two isomorphic subtrees should be drawn equally. Graph isomorphism means that there is a bijection between two graphs, so that any two nodes u and v are adjacent in the first graph if and only if their bijections are adjacent in the second graph.
- A tree and its mirror image should produce drawings that are reflections of one another.
- Integer coordinates should be preferred which leads to a grid drawing at the end.

Many tree layout algorithms use a divide and conquer strategy, such as the well-known Reingold/Tilford algorithm for binary trees [107]. In a postorder traversal of the tree, the following simple steps are executed:

1. Draw the left subtree.
2. Draw the right subtree.
3. Combine both drawings with a specific minimum distance.
4. Place the root of both subtrees at the next upper level exactly in the center of its subtrees.
5. In case the parent node has only one subtree, place the root in a specific horizontal distance.

Reingold/Tilford runs in linear time and can relatively easily be extended for the layout of general trees [13, 139]. Of course, there are further possibilities of drawing trees with the help of node-link diagrams, such as radial layouts, H-trees, or HV-trees. We refer the reader to the standard literature [24, 64]. Fig. 3 shows two example layouts computed with the *yED* tool [149].


(a) A standard tree layout for general trees.

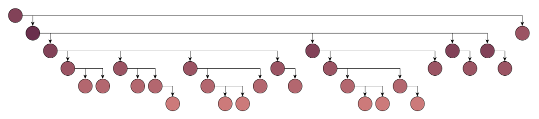

(b) A so-called HV-drawing in which the layout algorithm switches between the horizontal and vertical orientation.

Fig. 3 Two sample tree layouts that were computed and displayed by the *yED* graph editor [149]. The identical input tree has 30 nodes and 29 edges.

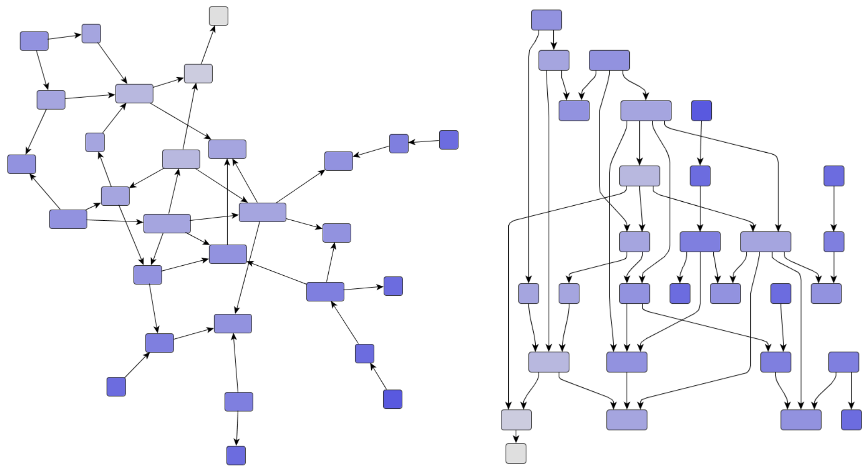
Force-based Drawings

Force-based layout techniques use a physical analogy to draw graphs and are widely used in practice. This is because of several reasons: the physical metaphor makes them easy to understand and to code, the results are suitable for many application fields, they are easy to extend with additional constraints, and the process of obtaining an equilibrium state (see below) can be animated which looks pretty nice. A simple version of a forced-based layout algorithm using spring and electrical repulsion forces is introduced in the following. Here, the edges between nodes are modeled as springs, and the nodes can be considered as charged particles that repel each other. For the x-component of the force vector on a node v the following holds (y-component analogous):

$$\sum_{(u,v)\in E} (sti_{uv}(d_{uv} - l_{uv}))\hat{x}_{uv} + \sum_{(u,v)\in V\times V} \frac{rep_{uv}}{d_{uv}^2}\hat{x}_{uv} \qquad (1)$$

Here, $\hat{x}_{uv}$ denotes the unit vector of $(x_v - x_u)$. d_{uv} is the Euclidean distance between u and v, l_{uv} is the zero-energy (natural) length of the spring between u and v (i. e., no force if $d_{uv} = l_{uv}$), $sti_{uv} \in [0,1]$ is the stiffness of the spring between u and v (i. e., the larger this parameter the more the tendency for d_{uv} to be close to l_{uv}), and finally rep_{uv} is the strength of the electrical repulsion between the two nodes. In Eq. 1, the first sum represents the spring force between two nodes u and v connected with an edge and the second sum the repulsion force between v and other nodes. Both forces together build a complete force system for all graph elements. Depending on the underlying physical model, the repulsion forces avoid that nodes are getting too close, and the spring forces provide a uniform edge length, for instance. In the current formula, Hook's law is used to specify the spring force between two nodes, i. e., if the distance between the two nodes is larger than the natural length of the spring, then the nodes attract each other. And the strength of the attraction is proportional to the difference between distance and natural length.

A simple algorithm that computes a final graph layout consists of a loop which firstly computes the forces of all nodes and then moves each node a bit into the direction of its force vector computed in Eq. 1. At the beginning, all nodes are positioned randomly. The loop is left if the sum of all forces together is small enough



(a) Result of a forced-based layout algorithm. (b) Layered (or hierarchical) drawing.

Fig. 4 Two sample graph layouts that were computed and displayed by the *yED* graph editor [149]. The identical input digraph has 29 nodes and 39 edges.

(equilibrium state) or after a specific number of iterations. This strategy works for undirected and directed graphs, with and without cycles, cf. Fig. 4(a).

Layered (Hierarchical) Drawings of Directed Graphs

A general aim for the layout of a directed graph is to compute a so-called *monotone* drawing in which all edges point into the same direction. Such a monotone drawing has some advantages in the interpretation of the digraph's topology [47]. Obviously, the input digraph must be acyclic in that case, otherwise we would get edges that flow backwards (called *feedback edges*). In practice this apparent hard condition is not really a problem, because we can use such a drawing method for general directed graphs if we change the direction of a minimal number of the feedback edges. This step is known as *cycle removal*. By doing so, we get a directed acyclic graph (DAG) that is drawn by using a method for computing monotone layouts, such as a layered drawing as explained in this paragraph. If the final layout is ready, we simply reverse the feedback edges again.

Many people prefer a hierarchical structure of the final graph layout, i. e., the nodes of the graph are arranged on vertical or horizontal, parallel layers in the 2D plane. Often, such a structure is already given by the input data. For instance, if someone wants to visualize hyperlinks (edges) between the HTML pages (nodes) of a website, then usually the pages are already hierarchically organized. In the following, we briefly present a standard technique for layered drawings that is based on the fundamental work of Sugiyama *et al.* [129].

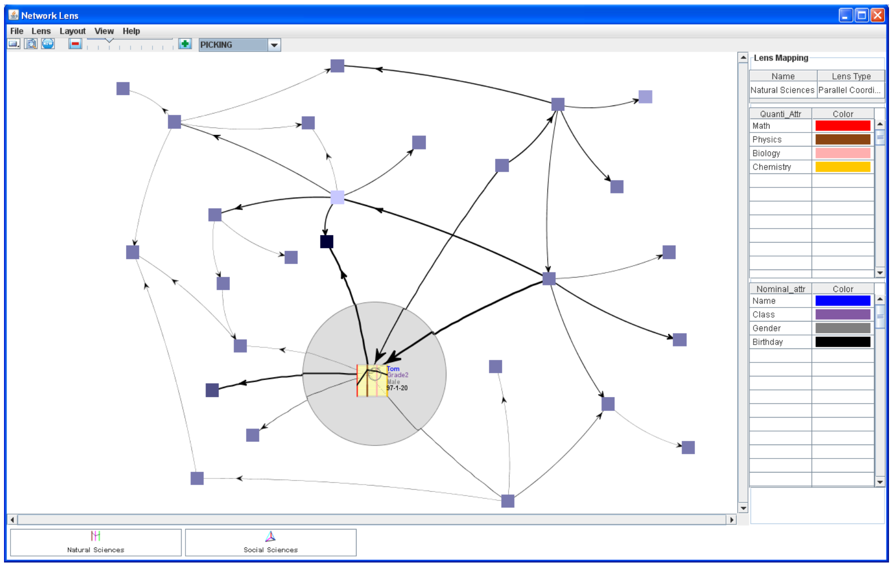
The basic idea is very simple and intuitive; it has three phases. In the first phase, the nodes of the graph are assigned to a number of layers (we can skip this phase if there is already a layering in the input graph). This layer assignment problem is NP-complete if we want to minimize the height and the width of the final layering. A further complication occurs if edges span over several layers: then we have to introduce so-called *dummy nodes* that lie on the spanned layers, i. e., a long edge is thus subdivided by the dummy nodes. This strategy causes modified edges which only reach from one layer to the next one (the digraph is called *proper* in such cases) and is needed for the second phase. After the layer assignment, we have to eliminate the number of edge crossings. This is done by reordering the graph nodes and the dummy nodes within each layer. With the help of the dummy nodes, the algorithm gets control over the edge positioning, and in consequence, it is possible to avoid crossings of edges that span over several layers. Minimizing edge crossings in a proper layered digraph is NP-complete, even if there are only two layers. Note that the node positions (x-coordinates) on the layers are relative only up to now (the y-coordinates of the nodes are already specified by the node layers if we assume to have horizontal layers). The final phase is the real coordinate assignment of all nodes on the layers, i. e., we assign concrete x-coordinates for each (normal and dummy) node. Also this task leads to an optimization problem that can be solved, for instance, by linear programming (LP). Constraints of the LP are then the fixed orderings in the layers, and the target function is specified by the straightness of the edges. As a final step, we remove the dummy nodes and obtain the wished layered drawing as shown in Fig. 4(b).

2.3 Multivariate Network Visualization

Good drawing algorithms as described in the previous subsection will not solely solve the problem of visualizing multivariate networks. There are several reasons for this statement. First, the most traditional graph drawings do not scale well, i. e., they are not able to represent huge data sets with many thousands of nodes and/or edges. Second, additional multivariate data cannot be intuitively embedded into a standard drawing. The InfoVis community tried to address those issues by visualization approaches that provide filtering and interaction possibilities in order to reduce the number of graph elements under consideration as well as by methods to visually analyze attributes in context of the underlying graph topology. Several approaches can be found in the literature that attempt to offer solutions for the problem of visualizing multivariate networks: *multiple and coordinated views*, *integrated approaches*, *semantic substrates*, *attribute-driven layouts*, and *hybrid approaches* [57]. We will discuss these concepts in the following paragraphs.

Multiple and coordinated Views: This category of solutions aims to combine several views and present them together. Coordinated views allow the use of the most powerful visualization techniques for each specific view and data set [41, 109]. As an application example, we highlight the work of Shanon *et al.* [120] who realized this idea in the network visualization domain. They use two distinct views: one view shows a parallel coordinate approach for the visual representation of the network attributes, and the other view displays a node-link drawing of a graph. Their tool is equipped with a variety of visualization and interaction techniques; both views are coordinated by linking and brushing [126] techniques. The drawback of multiple views is that they split the displayed data because of the spatial separation of the visual elements.

Integrated approaches: To provide a combined picture, attributes and the underlying graph can be displayed in one single view. "Integrated views can save space on a display and may decrease the time a user needs to find out relations; all data is displayed in one place." [41]. One example is described in Borisjuk's *et al.* [10] work on the visualization of experimental data in relation of a metabolic network. The authors used a straightforward approach by employing small diagrams instead of representing the nodes as simple circles or rectangles. Each diagram, e. g., a bar chart, shows experimental data that is related to the regarded node. This approach provides a view to all available information, but the embedding of the visualizations into the nodes causes the nodes to grow in size. This issue may affect the readability of the network due to the overlaps that may appear when



Lens Mapping

Name	Lens Type
Natural Sciences	Parallel Coordi...

Quanti_Attr	Color
Math	🟥
Physics	🟫
Biology	🟥
Chemistry	🟨

Nominal_attr	Color
Name	🟦
Class	🟪
Gender	⬜
Birthday	⬛

Fig. 5 Overview of the *Network Lens* tool [58]. The graphical user interface is divided into three distinctive parts: the main network visualization area, the lens information area on the right hand side, and the bottom part where user-produced lenses are preserved. It offers a way to visualize additional network attributes (displayed inside of the circular lens), while preserving the overall network topology and context. The lens in the screenshot covers one node only and shows a small parallel coordinate diagram with four quantitative as well as four nominal attributes belonging to that node. The user is able to move the lens with the mouse or to translate the graph behind the lens.

the number of nodes and the attributes is high [70]. Thus, is does not scale well. However, the problem of space usage and clutter introduced by such approaches can be avoided by using focus&context techniques (cf. Sect. 2.1). *Magic lenses* are one of several possibilities that are able to interactively visualize the node attributes within the same view as exemplified in Fig. 5.

Semantic substrates: In order to further avoid clutter in multivariate network visualizations, some researchers realized the idea of so-called semantic substrates that “are non-overlapping regions in which node placement is based on node attributes”: Shneiderman and Aris [122] introduced this idea and combined it with sliders to control the edge visibility and thus to ensure comprehensibility of the edges’ end nodes. One conceptual drawback of such approaches is that the underlying graph topology is not (completely) visible.

Attribute-driven layouts: Those layouts use the display of the network elements to present insight about the attached multivariate data instead of visualizing the graph topology itself. While being similar to semantic substrates, this technique does not necessarily place the nodes into specific regions. Instead, it uses calculations based on node attributes to control the placement of a node in the graph layout. An example is *PivotGraph* [142] which uses a grid-layout to show the relationship between (node) attributes and links.

Hybrid approaches: They combine at least two of the previously discussed techniques. The most common combinations are multiple coordinated views with any of the integrated approaches. For instance, Rohrschneider *et al.* [112] integrate additional attributes of a biological network inside the nodes and edges, see Fig. 6. The authors also use other visual metaphors for creating multiple coordinated views to show time-related data of the network.

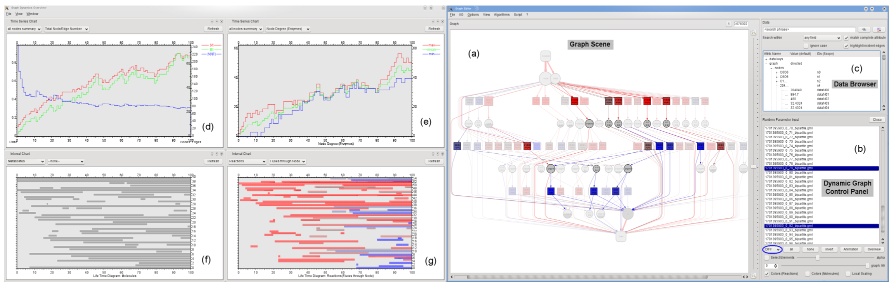
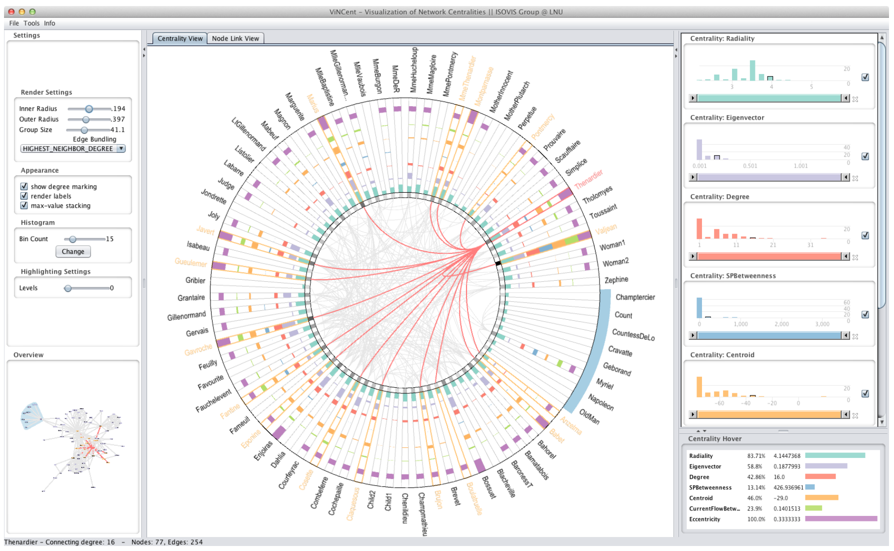


Fig. 6 The screenshot shows a tool for the visual analysis of dynamic metabolic networks [112]. On the left hand side, two time series charts of selected attributes display attribute dynamics over time. Interval charts represent the dynamic topology of the graph in terms of life times of metabolites, enzymes and reactions. On the right, the graph scene shows the set union graph (= the super graph that summarizes all nodes/edges of the individual graphs that appear over time) with the applied node coloring scheme which supports distinguishing between older and newer nodes.

2.4 Visual Analytics

Visual Analytics (VA) "is the science of analytical reasoning facilitated by interactive visual interfaces" [130]. A crucial property of this research field is that computational methods of data analysis are combined with interactive visualization techniques in order to analyze data more efficiently. *Automatic data analysis* covers various aspects from data storage and organization to automatic analysis algorithms, such as support vector machines, neural networks, PCA, etc. It might be classified among others into data management, data mining, and machine learning. For many data analysis problems, fully automated analysis methods only work for well-defined and well-understood problems, i. e., there has to exist a model of the underlying problem [65]. Otherwise, traditional data mining techniques will not work. Even if a model exists, then the results of the automated analyses have to be sufficiently communicated to and interpreted by analysts. Here, *interactive visualizations* come into the play as they are able to support the analyst to discover (possibly unexpected) patterns, trends, or relationships in the data. Interaction techniques (as presented in Sect. 2.1.3) are of particular importance to visually analyze large volumes of data. Interaction allows, among other things, to explore "unknown" data collections following Shneiderman's mantra of information visualization [121] or



Render Settings

Inner Radius .194
Outer Radius .397
Group Size 41.1
Edge Bundling HIGHEST_NEIGHBOR_DEGREE

Appearance

☑ show degree marking
☑ render labels
☑ max-value stacking

Histogram

Bin Count 15

Highlighting Settings

Levels 0

Radiality	83.71%	4.1447368
Eigenvector	58.8%	0.1877993
Degree	42.86%	16.0
SPBetweenness	13.14%	426.936961
Centroid	46.0%	-29.0
CurrentFlowBetw...	23.9%	0.1401513
Eccentricity	100.0%	0.3333333

Thenardier – Connecting degree: 16 – Nodes: 77, Edges: 254

Fig. 7 Overview of the *ViNCent* user interface [150]. The center shows the radial centrality view of the input network. The right side displays the corresponding histograms of the network centralities as well as detailed values of the network centralities for the currently hovered node. Histograms can be used to filter the views. The left panel allows changing the render settings and displays an overview of the respective node-link layout of the network. A node group has been manually selected and is shown as a light-blue stripe along the outer circle in the centrality view as well as in the overview (bottom left) by using a background region of the same color.

to build hypotheses with the help of “What if?”-questions and to verify them visually or with algorithmic methods. The need to combine interactive visualization with computational analysis methods is obvious and opens novel possibilities to address the information overload problem. A more detailed discussion on VA can be found in [65, 66, 130].

As an example from the field of visual network analysis, we have selected the *ViNCent* tool [72, 150], that combines exploratory data visualization with automatic analysis techniques, such as computing a variety of centrality values for network nodes as well as hierarchical clustering or node reordering based on centrality values. Automatic and interactive approaches are seamlessly integrated in one single analysis framework which provides insight into the importance of an individual node or groups of nodes and allows quantifying the network structure, see Fig. 7.

3 Visualization of Biological Networks

Visual representations of biological networks are widely used in the life sciences. Examples are shown in textbooks, on pathway posters, in databases and by a large number of tools for the analysis and visualization of biological processes. Well-known software tools are listed in Sect. 3.1.2. Software tools often use established layout methods as described in Sect. 2.2 to visualize biological networks automatically. Sometimes those algorithms are modified, for example, by adding extra forces to force-based approaches. However, often these methods do not or only partly take into account specific requirements for the visualization of a particular biological network and hence these visualizations are usually difficult to understand, especially if large networks are visualized.

In the following subsections, we will introduce some typical solutions for common networks from molecular biology, discuss domain-adapted solutions for particular networks, list major tools for the visualization of biological networks, and finally discuss the *Systems Biology Graphical Notation* (SBGN) as the graphical standard for biological networks.

3.1 Methods

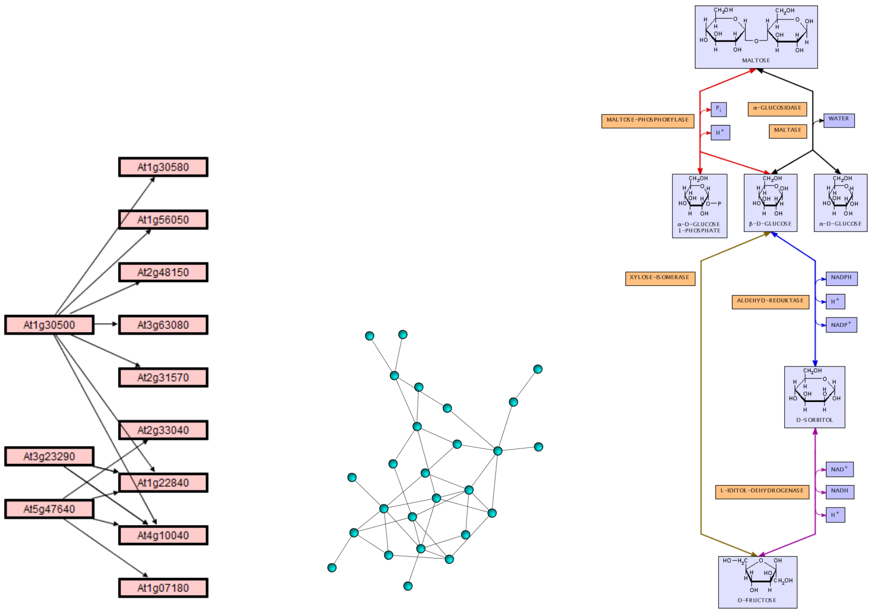
3.1.1 Early Approaches

Driven by the emerging availability of biological networks from databases in the mid-1990s, several groups started to either use existing graph drawing algorithms or designing extensions to these algorithms to automatically visualize biological networks. In the following, we present such early work for the three major types of networks from molecular biology.

Signal Transduction and Gene Regulatory Networks

These networks represent regulation or directed interaction between biological entities (such as genes) and are usually modeled as directed graphs, see Fig. 8(a). There are two widely used methods to visualize such networks: force-based and layered drawings. Several systems provide force-based graph drawing methods for the visualization of these networks, for example, *PATIKA* [23] and *GeNet* [118]. These tools typically use well-known force-based algorithms such as Eades' algorithm [28], often based on existing layout libraries and systems like *Pajek* [5] or *yFiles* [144]. There are some improvements of the general force-based method to consider application-specific requirements such as the representation of subcellular locations. One example is implemented in the *PATIKA* system.

Signal transduction and gene regulatory networks are directed graphs and, for example, the visualization of the main direction is important to understand the flow of information through the network. Therefore layered drawing methods are often employed for the computation of maps of these networks. Some tools using this



(a) A gene regulatory network (nodes represent genes, edges represent regulation, labels show gene names).

(b) A protein interaction network (nodes represent proteins, edges represent interaction).

(c) A metabolic network (nodes represent metabolites, enzymes, and reactions; edges represent consumption and production).

Fig. 8 Three sample layouts of biological networks. (a) and (b) were computed and displayed by the *Vanted* system [110]; (c) was computed by *BioPath* [33].

layout method are *TransPath* [85] and *BioConductor* [15]. Often layout libraries for layered drawings such as *dot* [84] are used.

Protein Interaction Networks

These networks represent proteins and their interactions and are modeled as undirected graphs, see Fig. 8(b). Several systems which employ force-based graph drawing methods for their visualization have been presented, for instance [12, 42, 98, 119]. Also some work on interactive exploration of protein interaction networks has been done, for example, by combining circular and force-based layouts and smooth transitions between subsequent drawings using animation [35].

Metabolic Networks

These networks represent the transformation of metabolites into each other and are usually modeled as directed graphs, see Fig. 8(c). There are two common approaches to visualizing metabolic networks: force-based and layered drawing methods. Several network analysis tools support force-based layouts, for example, *BioJAKE* [113], *Cytoscape* [119], *PathwayAssist* [101], and *VisANT* [46]. Frequently they visualize not only metabolic, but also other types of biological networks. However, force-based approaches mostly do not meet common application specific requirements. Such requirements are, inter alia, different sizes of nodes, the special placement of co-substances and enzymes, and the general direction of pathways.

Layered drawings are often used as they emphasis the main direction in the network. Tools supporting layered drawings are largely based on existing software libraries. Such solutions show the main direction within networks and partly deal with different node sizes. However, there is no specific placement of co-substances or special pathways such as cycles. Examples are *PathFinder* [40] (which uses the *VCG* library [114]) and *BioMiner* [123] (which employs *yFiles* [144]). The earliest approach to our knowledge is from Karp and Paley, where the complete network is separated into parts such as trees, paths and circles, and the parts are laid out separately [62]. Although not a layered drawing algorithm as described in Sect. 2.2, it results in an overall layout with some layered structure. Extended layered drawings consider cyclic structures within the network or show pathways of different topology using different layouts, such as the algorithm by Becker and Rojas [6]. An advanced layered drawing algorithm for metabolic networks considering all relevant visualization requirements has been presented in [115].

3.1.2 Current Approaches and Tools

There are many challenges in current research of biological network visualization and visual analytics, such as visual analysis of integrated and correlated data, visual comparison of networks, integrated and overlapping networks, graphical representation of paths and flows, and hierarchical networks, see [3, 39]. Consequently, this field has become very research active and, for example, several special algorithms have been presented in the last few years concerning the layout of biological networks. Among them are grid-based methods [81], clustered circular layouts [38], and constraint-based methods [117]. The quality of these specialized layout algorithms is often much better than just applying standard methods, an example is shown in Fig. 1.

A broad range of more than 170 tools for the modeling, analysis and visualization of biological networks is nowadays available on the Internet. These tools change often rapidly, new tools emerge, and old tools obtain new features or are not longer maintained. Therefore only a small set of some important tools will be listed here. Other reviews are available, for example, Suderman and Hallett in 2007 compared more than 35 tools regarding network and data visualization [128], Kono *et al.* compared tools for pathway representation, mapping and editing, and data exchange in 2009 [83], and Gehlenborg *et al.* looked at visualization tools for interaction networks and biological pathways in 2010 [39].

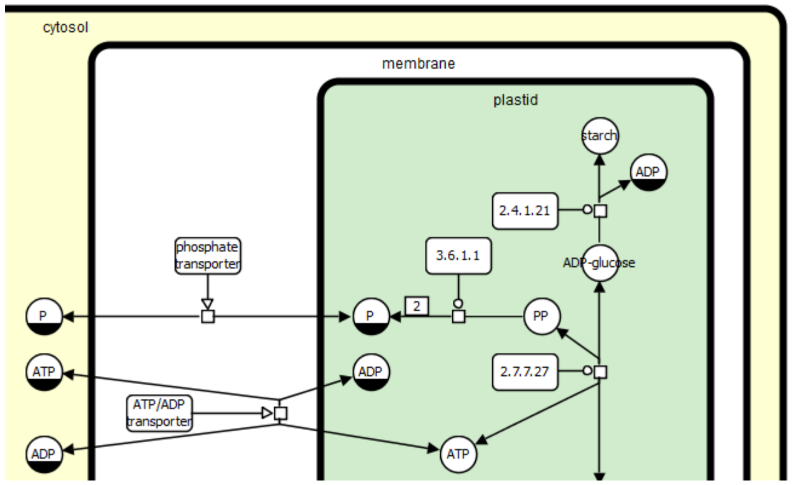
The following tools may be of interest to the reader. As the functionality of the tools changes rapidly over time, we do not provide a feature list but encourage the reader to visit the respective tool websites given below.

- *BiNa* [86] (`http://bit.ly/y6ix9i`)
- *BioUML* [82] (`http://bit.ly/yIETIt`)
- *CellDesigner* [36, 37] (`http://bit.ly/A0FQiF`)
- *CellMicrocosmos* [125] (`http://bit.ly/WJ8cnE`)
- *Cytoscape* [119, 124] (`http://bit.ly/wY2sbG`)
- *Omix* [26] (`http://bit.ly/zL52vB`)
- *Ondex* [78, Chapter 5] (`http://bit.ly/AetZjz`)
- *Pathway Projector* [83] (`http://bit.ly/zo5x2M`)
- *PathVisio* [135] (`http://bit.ly/zunwxW`)
- *Vanted* [56, 110] (`http://bit.ly/Aigr0T`)
- *VisAnt* [45, 46] (`http://bit.ly/agZBni`)

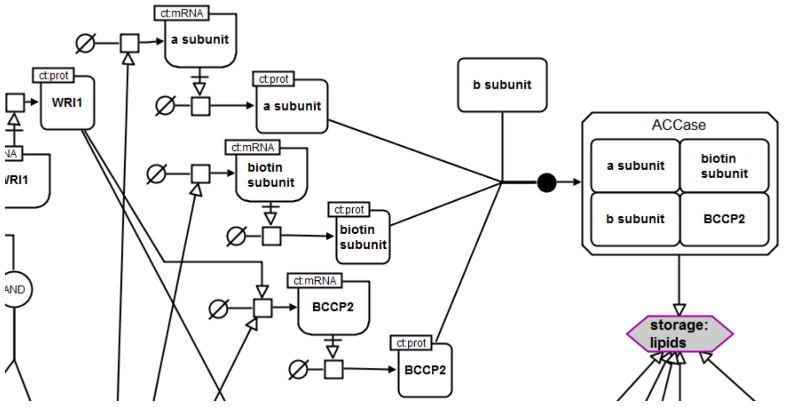
3.2 SBGN Standard

Biological networks shown in books, articles and online resources are often difficult to understand as the same biological concept can be shown by using different graphical representations. Therefore it is time consuming to get familiar with the graphical notation used, but this also carries the danger of misinterpretation. Consequently, particularly for molecular-biological networks such as gene regulatory, signal transduction, protein interaction and metabolic networks, there were several attempts to define a uniform representation. This includes Kitano's Process Diagrams [76], Kohn's Molecular Interaction Maps [79], and Michal's representation of metabolic pathways [95]. However, a single map type is often not enough to adequately illustrate the complexity of biological processes, and none of the mentioned attempts has asserted itself as a widely used standard.

Since 2006, there is a new initiative which partly builds on earlier standardization attempts and is closely connected with the successful exchange format *SBML* (System Biology Markup Language) [48]: *SBGN* – the System Biology Graphical Notation [87]. Additional material can be found under `http://sbgn.org`, and formal specifications are available [93, 97, 103], see the previously mentioned website for the latest version of the specification.



(a) Part of a metabolic pathway in SBGN notation (pathway derived from *MetaCrop* [116], an information system based on *Meta-All* [143]).



(b) Part of a gene regulatory network in SBGN notation (derived from *RIMAS* [54]).

Fig. 9 Two examples of SBGN maps.

SBGN supports three corresponding views or maps on a biological process: *Process Description* which describes elements (cellular building blocks like molecules, nucleic acid sequences, but also other information like observable events) and interactions between these elements; *Entity Relationship* which presents the interaction between biological entities and the influence of entities on other elements; and *Activity Flow* which focuses on the flow of information from one activity to another. These different language types enable to show different aspects of biological processes. A process description contains, for example, a molecule often several times in different states, e. g., phosphorylated or unphosphorylated, while both other map types show in each case only one occurrence of such a molecule. Fig. 9 shows two molecular-biological networks in SBGN notation.

There are several tools supporting SBGN, including *CellDesigner* [37], *EPE* (Edinburgh Pathway Editor) [30], *PathVisio* [135], and *SBGN-ED* [21] (an extension of *Vanted* [110]). A comparison has been done by Junker *et al.* [55]. There is also SBGN support for tool developers [136].

References

1. J. Abello and F. van Ham. Matrix zoom: A visual interface to semi-external graphs. In *Proceedings of the IEEE Symposium on Information Visualization*, pages 183–190, Los Alamitos, CA, USA, 2004. IEEE Computer Society.
2. W. Aigner, S. Miksch, H. Schumann, and C. Tominski. *Visualization of Time-Oriented Data*. Springer, 2011.
3. M. Albrecht, A. Kerren, K. Klein, O. Kohlbacher, P. Mutzel, W. Paul, F. Schreiber, and M. Wybrow. On open problems in biological network visualization. In *Proceedings of the International Symposium on Graph Drawing (GD '09)*, volume 5849 of *LNCS*, pages 256–267. Springer, 2010.
4. R. D. Appel, A. Bairoch, and D. F. Hochstrasser. A new generation of information retrieval tools for biologists: The example of the ExPASy WWW server. *Trends Biochemical Sciences*, 19:258–260, 1994.
5. V. Batagelj and A. Mrvar. Pajek – analysis and visualization of large networks. In M. Jünger and P. Mutzel, editors, *Graph Drawing Software*, pages 77–103. Springer, 2004.
6. M. Y. Becker and I. Rojas. A graph layout algorithm for drawing metabolic pathways. *Bioinformatics*, 17(5):461–467, 2001.
7. J. Beddow. Shape coding of multidimensional data on a microcomputer display. In *Proceedings of the 1st conference on Visualization '90*, VIS '90, pages 238–246, Los Alamitos, CA, USA, 1990. IEEE Computer Society Press.
8. J. M. Berg, J. L. Tymoczko, and L. Stryer. *Biochemistry*. W H Freeman, 2002.
9. C. Berge. *Hypergraphs: The Theory of Finite Sets*. North-Holland, Amsterdam, The Netherlands, 1989.
10. L. Borisjuk, M.-R. Hajirezaei, C. Klukas, H. Rolletschek, and F. Schreiber. Integrating data from biological experiments into metabolic networks with the DBE information system. *In Silico Biol*, 5(2):93–102, 2005.
11. M. Bostock. Edgar Anderson's Iris data set scatter plot matrix, last accessed: 2013-03-13. `http://mbostock.github.com/d3/talk/20111116/iris-splom.html`.
12. B. J. Breitkreutz, C. Stark, and M. Tyers. Osprey: a network visualization system. *Genome Biology*, 4(3):R22, 2003.

13. C. Buchheim, M. Jünger, and S. Leipert. Improving walker's algorithm to run in linear time. In *Revised Papers from the 10th International Symposium on Graph Drawing*, GD '02, pages 344–353, London, UK, UK, 2002. Springer-Verlag.
14. S. Card, J. Mackinlay, and B. Shneiderman, editors. *Readings in Information Visualization: Using Vision to Think*. Morgan Kaufmann, 1999.
15. V. J. Carey, J. Gentry, E. Whalen, and R. Gentleman. Network structures and algorithms in BioConductor. *Bioinformatics*, 21(1):135–136, 2005.
16. J. M. Chambers, W. S. Cleveland, B. Kleiner, and P. A. Tukey. *Graphical Methods for Data Analysis*. Belmont, CA: Wadsworth, 1983.
17. C. Chen. *Information Visualization. Beyond the Horizon*. Springer-Verlag, London Berlin Heidelberg, 2nd edition, 2004.
18. H. Chernoff. The use of faces to represent points in k-dimensional space graphically. *Journal of the American Statistical Association*, 68:361–368, 1973.
19. W. C. Cleveland and M. E. McGill. *Dynamic Graphics for Statistics*. CRC Press, Inc., Boca Raton, FL, USA, 1988.
20. C. Collins, F. Viegas, and M. Wattenberg. Parallel tag clouds to explore and analyze faceted text corpora. In *Proceedings of the IEEE Symposium on Visual Analytics Science and Technology (VAST '09)*, pages 91–98. IEEE Computer Society, 2009.
21. T. Czauderna, C. Klukas, and F. Schreiber. Editing, validating and translating of SBGN maps. *Bioinformatics*, 26(18):2340–2341, 2010.
22. D3. Data-Driven Documents, last accessed: 2013-03-13. `http://d3js.org`.
23. E. Demir, O. Babur, U. Dogrusöz, A. Gürsoy, G. Nisanci, R. Çetin Atalay, and M. Ozturk. PATIKA: an integrated visual environment for collaborative construction and analysis of cellular pathways. *Bioinformatics*, 18(7):996–1003, 2002.
24. G. Di Battista, P. Eades, R. Tamassia, and I. G. Tollis. *Graph Drawing: Algorithms for the Visualization of Graphs*. Prentice Hall, 1999.
25. S. Diehl, C. Görg, and A. Kerren. Preserving the mental map using foresighted layout. In D. S. Ebert, J. M. Favre, and R. Peikert, editors, *Data Visualization 2001*, Eurographics, pages 175–184. Springer Vienna, 2001.
26. P. Droste, S. Miebach, S. Niedenführ, W. Wiechert, and K. Nöh. Visualizing multi-omics data in metabolic networks with the software Omix: a case study. *Biosystems*, 105(2):154–161, 2011.
27. J. Dykes, A. M. MacEachren, and M.-J. Kraak. *Exploring Geovisualization*. Pergamon, 2005.
28. P. Eades. A heuristic for graph drawing. *Congressus Numerantium*, 42:149–160, 1984.
29. M. Eiglsperger, S. P. Fekete, and G. W. Klau. Orthogonal graph drawing. In M. Kaufmann and D. Wagner, editors, *Drawing Graphs*, volume 2025 of *Lecture Notes in Computer Science*, pages 121–171. Springer Berlin Heidelberg, 2001.
30. EPE. *EPE Edinburgh PAthway Editor*, last accessed: 2012-08-02. `http://epe.sourceforge.net/SourceForge/EPE.html`.
31. ExposeData.com. Nutrient Contents – Parallel Coordinates, last accessed: 2013-03-13. `http://exposedata.com/parallel/`.
32. J. Feinberg. Wordle, last accessed: 2013-03-13. `http://www.wordle.net`.
33. M. Forster, A. Pick, M. Raitner, F. Schreiber, and F. J. Brandenburg. The system architecture of the BioPath system. *In Silico Biology*, 2(3):415–426, 2002.
34. E. Freeman and S. Fertig. Lifestreams: Organizing your electronic life. In *AAAI Fall Symposium on AI Applications in Knowledge Navigation and Retrieval*, pages 38–44. Association for the Advancement of Artificial Intelligence, 1995.
35. C. Friedrich and F. Schreiber. Visualization and navigation methods for typed protein-protein interaction networks. *Applied Bioinformatics*, 2(3 Suppl):19–24, 2003.
36. A. Funahashi, Y. Matsuoka, A. Jouraku, H. Kitano, and N. Kikuchi. CellDesigner: a modeling tool for biochemical networks. In *Proceedings of the 38th conference on Winter simulation*, pages 1707–1712. Winter Simulation Conference, 2006.
37. A. Funahashi, M. Morohashi, and H. Kitano. CellDesigner: a process diagram editor for gene-regulatory and biochemical networks. *Biosilico*, 1(5):159–162, 2003.

38. D. Fung, M. Wilkins, D. Hart, and S. Hong. Using the clustered circular layout as an informative method for visualizing protein-protein interaction networks. *Proteomics*, 10(14):2723–2727, 2010.
39. N. Gehlenborg, S. I. O'Donoghue, N. S. Baliga, A. Goesmann, M. A. Hibbs, H. Kitano, O. Kohlbacher, H. Neuweger, R. Schneider, D. Tenenbaum, and A.-C. Gavin. Visualization of omics data for systems biology. *Nature Methods*, 7:S56–S68, 2010.
40. A. Goesmann, M. Haubrock, F. Meyer, J. Kalinowski, and R. Giegerich. PathFinder: reconstruction and dynamic visualization of metabolic pathways. *Bioinformatics*, 18(1):124–129, 2002.
41. C. Görg, M. Pohl, E. Qeli, and K. Xu. Visual Representations. In Kerren et al. [70], pages 163–230.
42. K. Han, B.-H. Ju, and J. H. Park. InterViewer: Dynamic visualization of protein-protein interactions. In S. G. Kobourov and M. T. Goodrich, editors, *Proceedings of the International Symposium on Graph Drawing (GD '02)*, volume 2528 of *LNCS*, pages 364–365. Springer, 2002.
43. J. Heer and B. Shneiderman. Interactive dynamics for visual analysis. *Communication of the ACM*, 55(4):45–54, Apr. 2012.
44. N. Henry, J.-D. Fekete, and M. J. McGuffin. Nodetrix: a hybrid visualization of social networks. *IEEE Transactions on Visualization and Computer Graphics*, 13:1302–1309, 2007.
45. Z. Hu, J.-H. Hung, Y. Wang, Y.-C. Chang, C.-L. Huang, M. Huyck, and C. DeLisi. VisANT 3.5: Multi-scale network visualization, analysis and inference based on the gene ontology. *Nucleic Acids Research*, 37(Web Server issue):W115–W121, 2009.
46. Z. Hu, J. Mellor, J. Wu, and C. DeLisi. VisANT: an online visualization and analysis tool for biological interaction data. *BMC Bioinformatics*, 5(1):e17, 2004.
47. W. Huang, P. Eades, and S.-H. Hong. A graph reading behavior: Geodesic-path tendency. In *Proceedings of the IEEE Pacific Visualization Symposium, 2009 (PacificVis '09)*, pages 137–144, April.
48. M. Hucka, A. Finney, H. M. Sauro, H. Bolouri, J. C. Doyle, H. Kitano, A. P. Arkin, B. J. Bornstein, D. Bray, A. Cornish-Bowden, A. A. Cuellar, S. Dronov, E. D. Gilles, M. Ginkel, V. Gor, I. I. Goryanin, W. J. Hedley, T. C. Hodgman, J. H. Hofmeyr, P. J. Hunter, N. S. Juty, J. L. Kasberger, A. Kremling, U. Kummer, N. Le Novere, L. M. Loew, D. Lucio, P. Mendes, E. Minch, E. D. Mjolsness, Y. Nakayama, M. R. Nelson, P. F. Nielsen, T. Sakurada, J. C. Schaff, B. E. Shapiro, T. S. Shimizu, H. D. Spence, J. Stelling, K. Takahashi, M. Tomita, J. Wagner, and J. Wang. The systems biology markup language (SBML): A medium for representation and exchange of biochemical network models. *Bioinformatics*, 19:524–531, 2003.
49. A. Inselberg and B. Dimsdale. Parallel coordinates: A tool for visualizing multi-dimensional geometry. In *Proceedings of the IEEE Conference on Visualization (Vis '90)*, pages 361–378. IEEE Computer Society, 1990.
50. W. Javed and N. Elmqvist. Exploring the design space of composite visualization. In *Proceedings of the IEEE Pacific Symposium on Visualization (PacificVis '12)*, pages 1–8. IEEE Computer Society Press, 2012.
51. D. F. Jerding and J. T. Stasko. The information mural: A technique for displaying and navigating large information spaces. *IEEE Transactions on Visualization and Computer Graphics*, 4(3):257–271, July 1998.
52. B. Johnson and B. Shneiderman. Tree-maps: a space-filling approach to the visualization of hierarchical information structures. In *Proceedings of the 2nd Conference on Visualization (Vis '91)*, pages 284–291, Los Alamitos, CA, USA, 1991. IEEE Computer Society Press.
53. I. Jolliffe. *Principal Component Analysis*. Springer Verlag, 2002.
54. A. Junker, A. Hartmann, F. Schreiber, and H. Bäumlein. An engineer's view on regulation of seed development. *Trends in Plant Science*, 15(6):303–307, 2010.
55. A. Junker, H. Rohn, T. Czauderna, C. Klukas, A. Hartmann, and F. Schreiber. Creating interactive, web-based and data-enriched maps using the systems biology graphical notation. *Nature Protocols*, 7:579–593, 2012.

56. B. H. Junker, C. Klukas, and F. Schreiber. VANTED: A system for advanced data analysis and visualization in the context of biological networks. *BMC Bioinformatics*, 7:109, 2006.
57. I. Jusufi. *Towards the Visualization of Multivariate Biochemical Networks*. Licentiate thesis, Linnaeus University, 2012.
58. I. Jusufi, Y. Dingjie, and A. Kerren. The network lens: Interactive exploration of multivariate networks using visual filtering. In *Proceedings of the 14th International Conference on Information Visualisation (IV '10)*, pages 35 –42. IEEE Computer Society Press, 2010.
59. I. Jusufi, A. Kerren, V. Aleksakhin, and F. Schreiber. Visualization of Mappings between the Gene Ontology and Cluster Trees. In *Proceedings of the SPIE 2012 Conference on Visualization and Data Analysis (VDA '12)*, SPIE 8294, pages 8294–20, Burlingame, CA, USA, 2012. IS&T/SPIE.
60. I. Jusufi, C. Klukas, A. Kerren, and F. Schreiber. Guiding the interactive exploration of metabolic pathway interconnections. *Information Visualization*, 11(2):136–150, 2012.
61. M. Kanehisa, S. Goto, S. Kawashima, and A. Nakaya. The KEGG databases at GenomeNet. *Nucleic Acids Research*, 30(1):42–46, 2002.
62. P. D. Karp and S. M. Paley. Automated drawing of metabolic pathways. In H. Lim, C. Cantor, and R. Bobbins, editors, *Proceedings of the International Conference on Bioinformatics and Genome Research*, pages 225–238, 1994.
63. O. Kaser and D. Lemire. Tag-Cloud Drawing: Algorithms for Cloud Visualization. In *Proceedings of Tagging and Metadata for Social Information Organization (WWW '07)*, Banff, Canada, 2007.
64. M. Kaufmann and D. Wagner. *Drawing Graphs: Methods and Models*, volume 2025 of *Lecture Notes in Computer Science Tutorial*. Springer, 1999.
65. D. Keim, G. Andrienko, J.-D. Fekete, C. Görg, J. Kohlhammer, and G. Melançon. Visual analytics: Definition, process, and challenges. In Kerren et al. [75], pages 154–175.
66. D. Keim, J. Kohlhammer, G. Ellis, and F. Mansmann, editors. *Mastering The Information Age – Solving Problems with Visual Analytics*. Eurographics Digital Library, 2010.
67. D. Keim and H.-P. Kriegel. Visdb: Database exploration using multidimensional visualization. *IEEE Computer Graphics and Applications*, 14(5):40–49, 1994.
68. D. Keim and D. Oelke. Literature Fingerprinting: A New Method for Visual Literary Analysis. In *Proceedings of the IEEE Symposium on Visual Analytics Science and Technology (VAST '07)*, pages 115–122, Sacramento, CA, USA, 2007. IEEE Computer Society Press.
69. D. A. Keim. Information visualization and visual data mining. *IEEE Transactions on Visualization and Computer Graphics*, 7(1):1–8, 2002.
70. A. Kerren, A. Ebert, and J. Meyer, editors. *Human-Centered Visualization Environments*. LNCS Tutorial 4417. Springer, 2007.
71. A. Kerren, A. Ebert, and J. Meyer. Introduction to Human-Centered Visualization Environments. In *Human-Centered Visualization Environments* [70], pages 1–9.
72. A. Kerren, H. Köstinger, and B. Zimmer. Vincent – visualisation of network centralities. In *Proceedings of the International Conference on Information Visualization Theory and Applications (IVAPP '12)*, pages 703–712. INSTICC, 2012.
73. A. Kerren and F. Schreiber. Toward the role of interaction in visual analytics. In *Proceedings of the Winter Simulation Conference*, WSC '12, pages 420:1–420:13. Winter Simulation Conference, 2012.
74. A. Kerren, J. T. Stasko, J.-D. Fekete, and C. North. Workshop report: Information visualizationhuman-centered issues in visual representation, interaction, and evaluation. *Information Visualization*, 6(3):189–196, 2007.
75. A. Kerren, J. T. Stasko, J.-D. Fekete, and C. North, editors. *Information Visualization: Human-Centered Issues and Perspectives*, volume 4950 of *Lecture Notes in Computer Science*. Springer, Berlin, Heidelberg, 2008.
76. H. Kitano. A graphical notation for biochemical networks. *Biosilico*, 1(5):169–176, 2003.
77. K. Koh, B. Lee, B. Kim, and J. Seo. Maniwordle: Providing flexible control over wordle. *IEEE Transactions on Visualization and Computer Graphics*, 16:1190–1197, 2010.

78. J. Köhler, J. Baumbach, J. Taubert, M. Specht, A. Skusa, A. Rüegg, C. Rawlings, P. Verrier, and S. Philippi. Graph-based analysis and visualization of experimental results with ONDEX. *Bioinformatics*, 22(11):1383–1390, 2006.
79. K. W. Kohn and M. I. Aladjem. Circuit diagrams for biological networks. *Molecular Systems Biology*, 2:e2006.0002, 2006.
80. T. Kohonen, M. R. Schroeder, and T. S. Huang, editors. *Self-Organizing Maps*. Springer-Verlag New York, Inc., Secaucus, NJ, USA, 3rd edition, 2001.
81. K. Kojima, M. Nagasaki, E. Jeong, M. Kato, and S. Miyano. An efficient grid layout algorithm for biological networks utilizing various biological attributes. *BMC Bioinformatics*, 8, 2007.
82. F. A. Kolpakov. BioUML – framework for visual modeling and simulation of biological systems. In *Proceedings of the International Conference on Bioinformatics of Genome Regulation and Structure*, pages 130–133. Springer, 2002.
83. N. Kono, K. Arakawa, R. Ogawa, N. Kido, K. Oshita, K. Ikegami, S. Tamaki, and M. Tomit. Pathway Projector: Web-based zoomable pathway browser using KEGG atlas and Google maps API. *PLoS ONE*, 4(11):e7710, 2009.
84. E. Koutsofios and S. North. Drawing graphs with dot. Technical report, AT&T Bell Laboratories, Murray Hill, NJ., 1995.
85. M. Krull, N. Voss, C. Choi, S. Pistor, A. Potapov, and E. Wingender. TRANSPATH: an integrated database on signal transduction and a tool for array analysis. *Nucleic Acids Research*, 31(1):97–100, 2003.
86. J. Küntzer, C. Backes, T. Blum, A. Gerasch, M. Kaufmann, O. Kohlbacher, and H.-P. Lenhof. Bndb - the biochemical network database. *BMC Bioinformatics*, 8:367, 2007.
87. N. Le Novère, M. Hucka, H. Mi, S. Moodie, F. Schreiber, A. Sorokin, E. Demir, K. Wegner, M. I. Aladjem, S. M. Wimalaratne, F. T. Bergman, R. Gauges, P. Ghazal, H. Kawaji, L. Li, Y. Matsuoka, A. Villéger, S. E. Boyd, L. Calzone, M. Courtot, U. Dogrusoz, T. C. Freeman, A. Funahashi, S. Ghosh, A. Jouraku, S. Kim, F. Kolpakov, A. Luna, S. Sahle, E. Schmidt, S. Watterson, G. Wu, I. Goryanin, D. B. Kell, C. Sander, H. Sauro, J. L. Snoep, K. Kohn, and H. Kitano. The Systems Biology Graphical Notation. *Nature Biotechnology*, 27(8):735–741, 2009.
88. B. Lee, N. Riche, A. Karlson, and S. Carpendale. Sparkclouds: Visualizing trends in tag clouds. *IEEE Transactions on Visualization and Computer Graphics*, 16(6):1182–1189, 2010.
89. Q. Li, X. Bao, C. Song, J. Zhang, and C. North. Dynamic query sliders vs. brushing histograms. In *CHI '03 Extended Abstracts on Human Factors in Computing Systems*, CHI EA '03, pages 834–835, New York, NY, USA, 2003. ACM.
90. J. Liu. Visualization of weather data: Temperature trend visualization, 2012. Bachelor's thesis, Linnaeus University, School of Computer Science, Physics and Mathematics, Växjö, Sweden.
91. S. MacNeil and N. Elmqvist. Visualization mosaics for multivariate visual exploration. *Computer Graphics Forum*, 2013.
92. K. V. Mardia. *Multivariate Analysis*. Academic Press, 1979.
93. H. Mi, F. Schreiber, N. L. Novère, S. Moodie, and A. Sorokin. Systems biology graphical notation: Activity flow language level 1. *Nature Precedings*, 2009.
94. G. Michal. *Biochemical Pathways (Poster)*. Boehringer Mannheim, 1993.
95. G. Michal. On representation of metabolic pathways. *BioSystems*, 47:1–7, 1998.
96. G. Michal. *Biochemical Pathways*. Spektrum Akademischer Verlag, 1999.
97. S. Moodie, N. L. Novère, A. Sorokin, H. Mi, and F. Schreiber. Systems biology graphical notation: Process description language level 1. *Nature Precedings*, 2009.
98. R. Mrowka. A java applet for visualizing protein-protein interaction. *Bioinformatics*, 17(7):669–670, 2001.
99. D. E. Nicholson. *Metabolic Pathways Map (Poster)*. Sigma Chemical Co., St. Louis, 1997.
100. F. Nightingale. *Notes on Matters Affecting the Health, Efficiency, and Hospital Administration of the British Army*. Harrison & Sons., 1858.

101. A. Nikitin, S. Egorov, N. Daraselia, and I. Mazo. Pathway studio – the analysis and navigation of molecular networks. *Bioinformatics*, 19(16):2155–2157, 2003.
102. M. Nöllenburg. Geographic Visualization. In Kerren et al. [70], pages 257–294.
103. N. L. Novère, S. Moodie, A. Sorokin, F. Schreiber, and H. Mi. Systems biology graphical notation: Entity relationship language level 1. *Nature Precedings*, 2009.
104. P. Oesterling, G. Scheuermann, S. Teresniak, G. Heyer, S. Koch, T. Ertl, and G. Weber. Two-stage framework for a topology-based projection and visualization of classified document collections. In *Proceedings of the IEEE Symposium on Visual Analytics Science and Technology (VAST '10)*, pages 91 –98. IEEE Computer Society, 2010.
105. J. O'Madadhain, D. Fisher, and T. Nelson. JUNG – Java Universal Network/Graph Framework, last accessed: 2013-01-27. `http://jung.sourceforge.net/`.
106. R. M. Pickett and G. G. Grinstein. Iconographic displays for visualizing multidimensional data. In *Proceedings of the 1988 IEEE International Conference on Systems, Man, and Cybernetics*, volume 1, pages 514–519, 1988.
107. E. M. Reingold and J. S. Tilford. Tidier drawing of trees. *IEEE Transactions on Software Engineering*, 7(2):223–228, 1981.
108. J. S. Richard, R. Catrambone, M. Guzdial, and K. Mcdonald. An evaluation of space-filling information visualizations for depicting hierarchical structures. *International Journal of Human-Computer Studies*, 53:663–694, 2000.
109. J. C. Roberts. Exploratory visualization with multiple linked views. In A. MacEachren, M.-J. Kraak, and J. Dykes, editors, *Exploring Geovisualization*. Elseviers, 2004.
110. H. Rohn, A. Junker, A. Hartmann, E. Grafahrend-Belau, H. Treutler, M. Klapperstck, T. Czauderna, C. Klukas, and F. Schreiber. VANTED v2: a framework for systems biology applications. *BMC Systems Biology*, 6(139), 2012.
111. M. Rohrschneider, C. Heine, A. Reichenbach, A. Kerren, and G. Scheuermann. A novel grid-based visualization approach for metabolic networks with advanced focus&context view. In *Proceedings of the International Symposium on Graph Drawing (GD '09)*, volume 5849 of *LNCS*, pages 268–279. Springer, 2010.
112. M. Rohrschneider, A. Ullrich, A. Kerren, P. F. Stadler, and G. Scheuermann. Visual network analysis of dynamic metabolic pathways. In *Proceedings of the 6th International Conference on Advances in Visual Computing – Volume Part I (ISVC '10)*, pages 316–327, Berlin, Heidelberg, 2010. Springer-Verlag.
113. W. Salamonsen, K. Y. Mok, P. Kolatkar, and S. Subbiah. BioJAKE: A tool for the creation, visualization and manipulation of metabolic pathways. In *Proceedings of the Pacific Symposium on Biocomputing*, pages 392–400, 1999.
114. G. Sander. Graph layout through the VCG tool. In R. Tamassia and I. G. Tollis, editors, *Proceedings of the DIMACS International Workshop on Graph Drawing (GD '94)*, pages 194–205. Springer, 1994.
115. F. Schreiber. High quality visualization of biochemical pathways in BioPath. *In Silico Biology*, 2(2):59–73, 2002.
116. 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.
117. F. Schreiber, T. Dwyer, K. Marriott, and M. Wybrow. A generic algorithm for layout of biological networks. *BMC Bioinformatics*, 10:375, 2009.
118. V. N. Serov, A. V. Spirov, and M. G. Samsonova. Graphical interface to the genetic network database GeNet. *Bioinformatics*, 14(6):546–547, 1998.
119. P. Shannon, A. Markiel, O. Ozier, N. S. Baliga, J. T. Wang, D. Ramage, N. Amin, B. Schwikowski, and T. Ideker. Cytoscape: a software environment for integrated models of biomolecular interaction networks. *Genome Research*, 13(11):2498–2504, 2003.
120. R. Shannon, T. Holland, and A. Quigley. Multivariate graph drawing using parallel coordinate visualisations. Technical Report 2008-6, University College Dublin, School of Computer Science and Informatics, 2008.

121. B. Shneiderman. The eyes have it: A task by data type taxonomy for information visualizations. In *Proceedings of the IEEE Symposium on Visual Languages (VL '96)*, pages 336–343. IEEE Computer Society, 1996.
122. B. Shneiderman and A. Aris. Network visualization by semantic substrates. *IEEE Transactions on Visualization and Computer Graphics*, 12:733–740, 2006.
123. M. Sirava, T. Schäfer, M. Eiglsperger, M. Kaufmann, O. Kohlbacher, E. Bornberg-Bauer, and H.-P. Lenhof. BioMiner – modeling, analyzing, and visualizing biochemical pathways and networks. *Bioinformatics*, 18(Suppl. 2):S219–S230, 2002.
124. M. E. Smoot, K. Ono, J. Ruscheinski, P.-L. Wang, and T. Ideker. Cytoscape 2.8: new features for data integration and network visualization. *Bioinformatics*, 27(3):431–432, 2011.
125. B. Sommer, J. Künsemöller, N. Sand, A. Husemann, M. Rumming, and B. Kormeier. Cellmicrocosmos 4.1 - an interactive approach to integrating spatially localized metabolic networks into a virtual 3d cell environment. In A. L. N. Fred, J. Filipe, and H. Gamboa, editors, *Proceedings of the First International Conference on Bioinformatics (BIOINFORMATICS '10), Valencia, Spain*, pages 90–95, 2010.
126. R. Spence. *Information Visualization: Design for Interaction*. Prentice Hall, 2nd edition, 2007.
127. J. Stasko and J. Muthukumarasamy. Visualizing program executions on large data sets. In *Proceedings of the IEEE Symposium on Visual Languages (VL '96)*, pages 166–173. IEEE Computer Society, 1996.
128. M. Suderman and M. T. Hallett. Tools for visually exploring biological networks. *Bioinformatics*, 23(20):2651–2659, 2007.
129. K. Sugiyama, S. Tagawa, and M. Toda. Methods for visual understanding of hierarchical system structures. *IEEE Transactions on Systems, Man and Cybernetics*, SMC-11(2):109–125, 1981.
130. J. J. Thomas and K. A. Cook. A visual analytics agenda. *IEEE Computer Graphics and Applications*, 26(1):10–13, 2006.
131. E. R. Tufte. *Envisioning Information*. Graphics Press, Cheshire, CT, USA, 1990.
132. E. R. Tufte. *Visual Explanations: Images and Quantities, Evidence and Narrative*. Graphic Press, Cheshire, CT, USA, 1997.
133. E. R. Tufte. *The Visual Display of Quantitative Information*. Graphics Press, Cheshire, CT, USA, 2nd edition, 2001.
134. F. van Ham and J. J. van Wijk. Beamtrees: compact visualization of large hierarchies. In *Proceedings of the IEEE Symposium on Information Visualization (InfoVis '02)*, pages 93–100. IEEE Computer Society, 2002.
135. M. P. van Iersel, T. Kelder, A. R. Pico, K. Hanspers, S. Coort, B. R. Conklin, and C. Evelo. Presenting and exploring biological pathways with PathVisio. *BMC Bioinformatics*, 9:399.1–9, 2008.
136. M. P. van Iersel, A. Villéger, T. Czauderna, S. E. Boyd, F. T. Bergmann, A. Luna, E. Demir, A. A. Sorokin, U. Dogrusöz, Y. Matsuoka, A. Funahashi, M. I. Aladjem, H. Mi, S. L. Moodie, H. Kitano, N. L. Novère, and F. Schreiber. Software support for SBGN maps: SBGN-ML and LibSBGN. *Bioinformatics*, 28(15):2016–2021, 2012.
137. J. J. Van Wijk and W. A. A. Nuij. Smooth and efficient zooming and panning. In *Proceedings of the IEEE Conference on Information Visualization (InfoVis '03)*, pages 15–22, Washington, DC, USA, 2003. IEEE Computer Society.
138. F. B. Viegas, M. Wattenberg, and J. Feinberg. Participatory visualization with wordle. *IEEE Transactions on Visualization and Computer Graphics*, 15:1137–1144, 2009.
139. J. Q. Walker. A node-positioning algorithm for general trees. *Softw. Pract. Exper.*, 20(7):685–705, July 1990.
140. M. Ward, G. Grinstein, and D. A. Keim. *Interactive Data Visualization: Foundations, Techniques, and Application*. A.K. Peters, Ltd., 2010.
141. C. Ware. *Information Visualization: Perception for Design*. Morgan Kaufmann, 2nd edition, 2004.

142. M. Wattenberg. Visual exploration of multivariate graphs. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems (CHI '06)*, pages 811–819, New York, NY, USA, 2006. ACM.
143. S. Weise, I. Grosse, C. Klukas, D. Koschützki, U. Scholz, F. Schreiber, and B. H. Junker. Meta-All: a system for managing metabolic pathway information. *BMC Bioinformatics*, 7:465, 2006.
144. R. Wiese, M. Eiglsperger, and M. Kaufmann. yFiles: Visualization and automatic layout of graphs. In P. Mutzel, M. Jünger, and S. Leipert, editors, *Proceedings of the International Symposium on Graph Drawing (GD '01)*, volume 2265 of *LNCS*, pages 453–454. Springer, 2001.
145. M. Williams and T. Munzner. Steerable, progressive multidimensional scaling. In *Proceedings of the IEEE Symposium on Information Visualization (InfoVis '04)*, pages 57–64. IEEE Computer Society Press, 2004.
146. C. Williamson and B. Shneiderman. The dynamic homefinder: evaluating dynamic queries in a real-estate information exploration system. In *Proceedings of the International ACM Conference on Research and Development in Information Retrieval (SIGIR '92)*, pages 338–346, New York, NY, USA, 1992. ACM.
147. J. Wise, J. Thomas, K. Pennock, D. Lantrip, M. Pottier, A. Schur, and V. Crow. Visualizing the non-visual: Spatial analysis and interaction with information from text documents. In *Proceedings of the IEEE Symposium on Information Visualization (InfoVis '95)*, pages 51–58. IEEE Computer Society, 1995.
148. J. S. Yi, Y. a. Kang, J. Stasko, and J. Jacko. Toward a deeper understanding of the role of interaction in information visualization. *IEEE Transactions on Visualization and Computer Graphics*, 13(6):1224–1231, 2007.
149. yWorks. *yEd Graph Editor*, last accessed: 2012-08-02. `http://www.yworks.com/en/products_yed_about.html`.
150. B. Zimmer, I. Jusufi, and A. Kerren. Analyzing multiple network centralities with ViNCent. In *Proceedings of SIGRAD 2012: Interactive Visual Analysis of Data, November 29-30, 2012, Växjö, Sweden,*, number 81 in Linköping Electronic Conference Proceedings, pages 87–90. Linköping University Electronic Press, 2012.