Visualization of modular structures in biological networks

Visualization of Modular Structures in Biological Networks

PROEFSCHRIFT

ter verkrijging van de graad van doctor aan de Technische Universiteit Eindhoven, op gezag van de rector magnificus prof.dr.ir. F.P.T. Baaijens, voor een commissie aangewezen door het College van Promoties, in het openbaar te verdedigen op dinsdag 1 september 2015 om 16:00 uur

door

Kasper Dinkla

geboren te Eindhoven

Dit proefschrift is goedgekeurd door de promotoren en de samenstelling van de promotiecommissie is als volgt:

This research was supported by the Netherlands' Organization for Scientific Research (NWO) as part of the Spotter project. NWO grant number 612.001.004.

This work was carried out in the ASCI graduate school. ASCI dissertation series number 328.

A catalogue record is available from the
Eindhoven University of Technology Library

ISBN	978-90-386-3888-1
printing	Ipskamp Drukkers
cover design	Kasper Dinkla

Preface

I have taken many things for granted while growing up in Eindhoven: a stable home, high-tech equipment at every corner, a top-tier but nonetheless inclusive education system, and a large number of people who are hell-bent on pushing boundaries. While the world economy went dark, the city of light doubled down on research and development. These good fortunes have helped me considerably in writing this thesis and I have many personal acknowledgements to give. At first I considered drawing a subdivided map of all the people that I want to give thanks to, but I know better after five years of designing visualizations. A table that consists of categorized names and words of thanks is clearly more effective from a usability standpoint:

As a	I thank	for
former doctoral candidate	Danny Niels Jing	getting me up to speed as a new doctoral candidate.
fellow doctoral candidate	Mickael Roeland Stef	fetching many cups of coffee with me, at 15:00 sharp, and enjoying many jokes at the progress bar level.
upcoming doctoral candidate	Alberto Bram Paul	bringing a breath of fresh air.
colleague	Andrei Huub	sharing your educational experience during course instructions.
secretary	Meivan	teaching me office essentials, such as the art of operating a fax machine.
pupil	Gijs Natalia	giving me the opportunity to supervise your graduation projects.
MSR colleague	Nathalie Yann Sanjay	your hospitality at Microsoft Research and introducing me to American peculiarities like decorating gingerbread houses.
friend	Hendrik	opening doors and pointing out the beer at conferences.

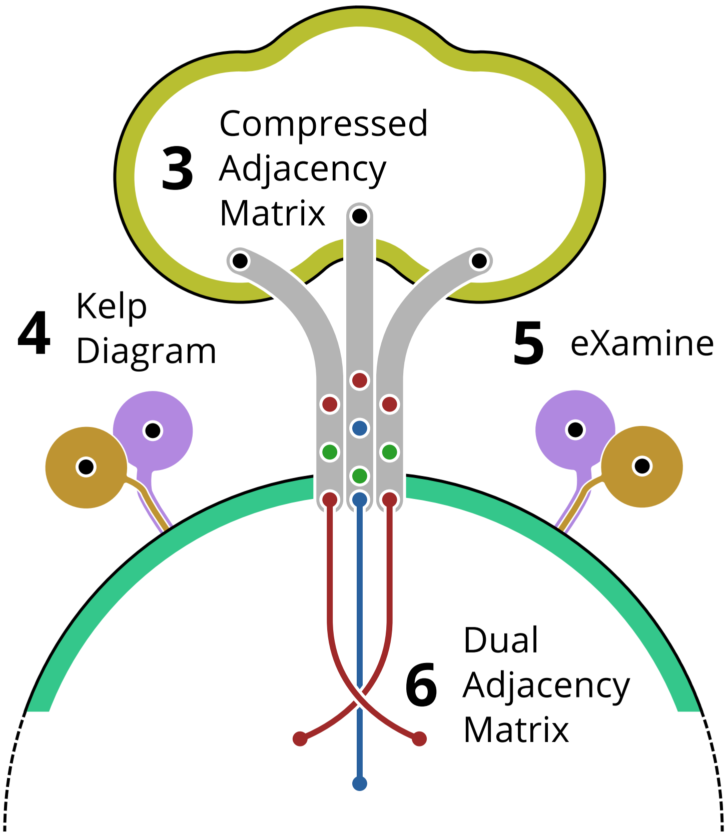
As a	I thank	for
collaborator	Bettina Mark	showing me the ropes of applied geometric algorithms and cartography.
	Gunnar Mohammed	exposing me to the more involved aspects of bioinformatics and your relentless enthusiasm.
promotor	Jack	always being able to free up time for your students, in spite of your capacities as a full professor and vice dean. These days your brainstorm sessions of 2+ hours are lovingly referred to as “being Jacked”.
copromotor	Michel	introducing me to the frontier of biological visualization and your subsequent trust in my ability as a student and doctoral candidate. This thesis is the result of your enthusiasm and tolerance of my creative process.
brother	Sip	sharing my journey as a (post-)doctoral candidate in good spirits and company, Nuria in particular. I remember sitting next to you on a Saturday morning when I received my first paper rejection. You cheered me up with loud laughter: “...and now you know how it feels.”
father	Ernst	teaching me the core principles of engineering and design as a child, which I ignored for many subsequent years. Too few children have the privilege to partake in dinner table discussions about two-stroke engines and 19th century European politics in short succession.
mother	Ria	encouraging my creative activities as a child by teaching me about sketching, painting, and composition, which I ignored for many subsequent years. I admire your tolerance of dinner table discussions about two-stroke engines and 19th century European politics.

And to anyone who is not named: Thank you.

Contents

1

Introduction



Many of the processes known to take place in biological cells are analyzed in the form of different types of networks. Network visualizations form a significant part of the biologist's analysis toolkit, from static wall posters [1] to interactive platforms such as Cytoscape [2]. These methods are elementary, generic, and enable rapid analysis of diverse data sets.

The size and complexity of biological networks increases along with our knowledge of cell processes, which impedes their analysis with existing, generic methods. This complexity is in part caused by a rich biological context. Additional structure can therefore be imposed on, or derived from, biological networks in the form of modules, which are subnetworks that are likely associated with specific cell processes. In this work we introduce, demonstrate, and discuss techniques that aim to support biological network analysis via the visualization of modular structures. These techniques are described in self-contained chapters:

Background In Chapter 2 we first describe three network types that are commonly used in analysis of cell biology: gene regulatory, protein interaction, and metabolic networks. Visualization practices, problems, and opportunities are discussed as a precursor to subsequent chapters, which (amongst others) concern all three network types and their modular structures. In addition, we provide insight into the complexity encountered in cell biology by discussing the need to visually integrate the three network types and enabling comparison of networks that belong to different organisms.

Compressed Adjacency Matrix Gene regulatory networks have structural characteristics that impede standard visualization techniques such as node-link diagrams and adjacency matrices. However, in Chapter 3 we show how these characteristics can be leveraged to cut open, rearrange, and compress adjacency matrices. These compressed matrices enable easy, visual detection of structures that are commonly associated with specific regulatory functions. We discuss those structures that are of interest to analysts, and relate them to network analysis tasks common to the visualization domain. Analysts can easily find these structures in compressed adjacency matrices, while the same is hard in standard adjacency matrix and node-link diagrams. This shows how gene regulatory networks allow for visualizations that emphasize specific structures while avoiding explicit emphasis via annotations.

Kelp Diagrams Some situations merit explicit emphasis of structures, such as highlighting a region of interest on top of an existing metabolic pathway visualization. When dealing with a pre-positioned node-link diagram, delimiting structures with contours is an obvious approach but one with visual caveats. These caveats become apparent in Chapter 4, in which Kelp Diagrams are introduced. These diagrams are capable of highlighting multiple overlapping structures (or set relations) on top of pre-positioned nodes. Kelp Diagrams consist of schematic contours that are generated with special attention to aesthetic quality, efficiency, and effectiveness. Comparison of Kelp Diagrams to two existing methods demonstrates the relevance of these concerns and the difficulty of visualizing complex structures, even for small data sets.

eXamine If overlapping structures are hard to emphasize as an addition to smaller networks such as metabolic pathways, then visualizing it for larger Protein Interaction Networks should be an even greater challenge. Conveniently, biologists are specialized in specific cell processes, types, and conditions. Biologists therefore often require the analysis of but small parts of the larger Protein Interaction Networks. Such a small part, or module, can be extracted because of recent advances that combine methods from statistics and combinatorics. Commonly, a multitude of set-based annotations are then attached to the module via statistical analysis with readily available knowledge such as Gene Ontology terms.

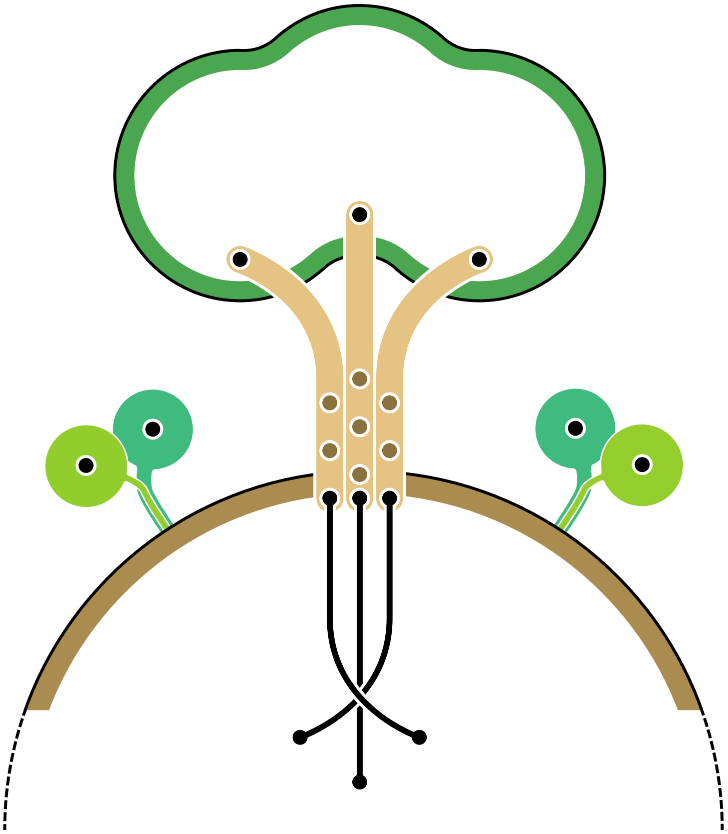
We present eXamine in Chapter 5, which is a Cytoscape app that supports the analysis of annotated modules. It integrates Kelp Diagram methodology into a simple layout algorithm and enables interactive exploration of a module and its annotations. We show the added value of this methodology via a case study that has resulted in a biological insight regarding the protein US28, which is associated with the development of certain types of cancer.

Dual Adjacency Matrix Subnetworks are often interpreted as a group of nodes accompanied by a group of links that connect them, also known as an induced subnetwork. This natural perspective is pervasive in aforementioned chapters, but in Chapter 6 we explore the alternate perspective; grouped links that are connected by nodes. We introduce the Dual Adjacency Matrix, which combines link and node group techniques such that an analyst can navigate the conceptual gap between link and node groups. Enabling the analysis of link groups is of particular interest for dense, or total, networks that contain additional link information, such as brain connectivity networks that are obtained via functional MRI studies.

Retrospective The mentioned chapters describe each technique in isolation, often skipping details of the underlying research process, pitfalls, and surprises. These details are left out to maintain clarity and focus, but can provide important insights nonetheless. In Chapter 7 we therefore conclude with a discussion about these issues and how they fit into the bigger picture of biological network visualization.

2

Background



K. Dinkla and M.A. Westenberg

Parts of this chapter have been published in Tsinghua Science and Technology [210].

2.1. Introduction

Understanding cell biology is key to the development of effective medicines and, therefore, subject to large-scale research. A biological cell consists of chemical compounds and its behavior is determined by interactions and reactions between these compounds. These reactions and interactions are often represented by three types of abstract networks:

Gene Regulatory Network Genes and how they promote or inhibit each other;

Protein Interaction Network Proteins and their affecting or binding interactions;

Metabolic Network Metabolites and metabolite transforming reactions.

These networks are derived from experiments on cells and existing knowledge of chemistry, but are still incomplete and imprecise. Networks become more complete as technology advances and therefore grow in complexity, which poses new analysis challenges.

Network visualization has been subject to much research and is pervasive in software tools used by the bioinformatics community. However, the mentioned network types have proven themselves hard to visualize and no decisive solutions have emerged from the bioinformatics or visualization communities. We provide an overview of the challenge of visualizing these networks, which stems from both the structure of the networks and the needs of the scientists that analyze them. Instead of focusing on visualization *tools* from the target domain [3, 4], a more generic perspective is given, focusing on *techniques* already in use and those that could be used. We first provide an overview of network visualization in general, after which we dive into the more concrete biological network types. Finally, we discuss the integration of these networks, and network alignment.

We describe some of the workings of biological cells to explain the semantics of biological networks and their role in research. However, we focus on the visualization of networks and not on biological nuance.

2.2. Network Visualization

Much research in the visualization community has focused on networks [5, 6], and the structurally more specific trees [7]. The node-link diagram and adjacency matrix are the prime visualization techniques, as shown in Fig. 2.1.

Analysts who use networks often have specific analysis goals that can be expressed as queries that in turn are answered through computation. Knowledge of specific analysis goals leads to the development of more effective visualizations. Since these goals are often ill defined, an alternative approach is to consider basic questions or tasks that underly a global analysis goal [8, 9]. The same approach is taken for the techniques that are introduced in subsequent chapters.

Most people are able to interpret node-link diagrams without receiving further explanation. Visualization tools for biological networks, such as Cytoscape [2, 10], VisANT [11], BiologicalNetworks [12, 13], VANTED [14], and GENeVis [15], therefore often use this representation. However, this intuitiveness can be overshadowed by the visual clutter of many intersecting links. Dense networks suffer from this problem, but also sparse networks with dense subnetworks.

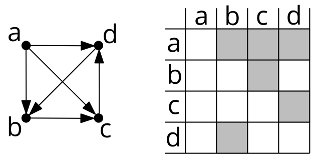


Figure 2.1: *Examples of the two dominant network visualization techniques: node-link diagram (left) and adjacency matrix (right). A gray cell indicates the presence of a link.*

Drawing node-link diagrams according to aesthetic criteria, and the combinatorics involved, is the focus of the *Graph Drawing* domain [16, 17]. Important aesthetic criteria involve visual clutter (e.g., node and link intersections) and the expression of network structure (e.g., path distance and symmetries). Force simulation is the most often used technique to lay out node-link diagrams, because it is simple and effective [18, 19]. More advanced techniques analyze network structure and express important structures in the layout [20], or give links more elaborate geometry than straight lines [21, 22].

Adjacency matrices are a potent alternative to node-link diagrams when the network to depict is dense, because they avoid the clutter of intersecting lines by encoding links as markers. It also enables the use of *rearrangement clustering* [23] to arrange the rows and columns of the matrices in such a way that clusters and patterns are more easily detected. Adjacency matrices are seldom used in biology to depict networks, though they are pervasive in the form of weight matrices and corresponding heat map representations that derive from microarray experiments [24]. Such heat maps are often combined with biology-specific rearrangement clustering [25]. Adjacency matrices use a complex, unfamiliar encoding, in which paths are difficult to follow. This may explain why adjacency matrices are uncommon in the biological domain.

Interaction methods [26] alleviate the analysis of networks that are hard to visualize. Nodes and links that are not of immediate interest can be *filtered* out to reduce clutter. Subnetworks, such as the neighborhood of a node, are sometimes *highlighted* to stand out from the rest of the network. Warping the space around a subnetwork is another way to emphasize it [27].

Many of the more advanced techniques introduced by the visualization community have not been adopted by the bioinformatics community. This in part has to do with the lack of knowledge about such techniques in bioinformatics and the effort required to implement complex algorithms for possibly little gain. Another reason is the long tradition of using manually drawn schematics that are not suited for integration with existing generic techniques. Most biological network specific tools therefore

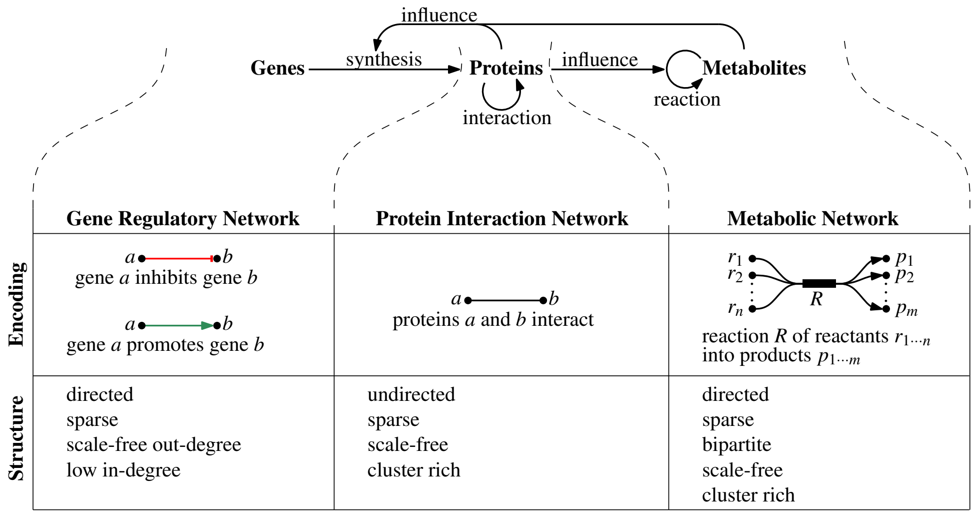


Figure 2.2: *Overview of the three main network types used in biological cell analysis.*

have many integrated analytic algorithms but few visualization options.

2.3. Networks in Cell Biology

The chemical processes within biological cells are studied with three types of network: the Gene Regulatory Network, Protein Interaction Network, and Metabolic Network. The following sections describe each type in detail. First, however, it is important to realize that these types, though presented as separate, describe cell processes that are strongly intertwined. Figure 2.2 gives an overview of the major components of cell processes and how they relate to each other. It also shows how each network type concerns the (indirect) interplay between compounds of a specific type, e.g., gene regulatory networks on genes and how they influence each other.

All of the discussed network types are *scale-free*. A network is said to be scale-free when the degree distribution of its nodes can be (partially) described by a power law, such that there are few nodes with a high degree and many nodes with a low degree. The origin of this property is commonly associated with the growth process of these networks as species evolve [28] and makes a network error tolerant [29]. It has been shown that scale-free networks are *small-world* [30] and therefore consist of many nodes that can be reached from any other node with a shortest path that spans few nodes.

2.3.1. Gene regulatory networks

The cell's chromosome(s) consists of a large number of blueprints for the construction of *proteins*, which are the building blocks and workers of the cell. Each section of a chromosome that is a viable protein blueprint, or performs another active role, is referred to as a *gene*. The process of constructing a protein from a gene consists of *transcription* and subsequent *translation*, summarized as *synthesis* in Fig. 2.2. The rate at which a gene is transcribed is called its *expression* and affects the rate of protein synthesis. The expression of a gene may affect the conditions within the cell and these conditions can subsequently affect the expression of other genes. Thus, the expression of one gene may influence the expression of another, which is called *regulation*.

The physical complexity of regulation, and the involvement of proteins and metabolites, is usually abstracted from, such that only two types of regulation between two genes are left; the expression of a gene either *promotes* or *inhibits* the expression of another gene. Though crude, this kind of information provides important insight into the roles of different genes in a cell. For example, a gene could act as a master switch, where its expression leads to the expression of many other genes.

A set of genes and known regulations between genes form a *Gene Regulatory Network* (or GRN). There are various ways to infer a GRN. For example, regulations can be derived statistically from large-scale experiments, where the expression of many genes is measured for varying cell conditions [31]. Mathematical modeling and simulation is also used to verify or predict regulations [32]. A GRN can also be composed from results reported in multiple papers that pinpoint individual regulations [33].

Structure There are several important structural characteristics that, in general, apply to a GRN [28, 34]: (1) Node in-degrees have a low average with little deviation and the network is therefore sparse; (2) The network is scale-free with respect to out-degree; (3) There are few cycles (excluding self-cycles) and every cycle is short.

GRNs are small-world because of their scale-free structure and are layered because of the presence of few and localized cycles, in which regulatory signals are passed from few genes at top layers down to many genes at deeper layers. The few genes with high out-degree (which tend to be at top layers) are known as *regulators* and their outbound neighborhoods as *regulons*.

Certain subnetwork structures are statistically overrepresented in GRNs with respect to networks with a random structure [35]. These subnetworks are called *motifs*, shown in Fig. 2.3, and are associated with specific regulatory behavior. The *auto-regulation* motif consists of a gene that regulates itself, which affects its sensitivity to regulations from other genes. The *feed-forward loop* consists of three genes, with one direct and one indirect regulation, that enable the filtering or creation of short regulation signals. In a *single-input module* motif the genes of a regulon are affected by only one regulator, such that multiple (possibly sequential) gene activations occur. The *multiple-input module* motif (or *dense overlapping regulons*) extends the single-input module to include multiple regulons with strong overlap, which enable different sequences of activation for the same genes.

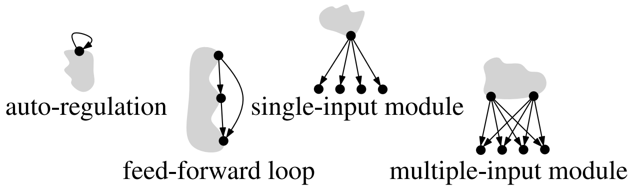


Figure 2.3: *Illustrations of GRN motifs. Gray areas represent network remainders.*

Analysis Analysis of GRNs concerns both global and local structures [36]. The global structure of the network indicates to what extent it is vulnerable to gene alterations [29]. For example, if the function of a cell is to be changed by the manipulation of a gene, it is useful to predict the global effect of its alteration. This relates to the common network tasks of determining a node's neighborhood and finding reachable nodes (see Section 2.2). The local structure provides detail about more specific cell behavior, like the behavior that is associated with motifs.

Visualization practices Node-link diagrams laid out through force simulation are most commonly used for GRN analysis. This results in hard to interpret visualizations for large GRNs due to the network's scale-free structure, as shown in Fig. 2.4. These visualizations are nonetheless used for analysis and accompany published results [37]. Smaller GRNs are drawn manually by biologists, placing emphasis on key areas of the network. The layered structure of a GRN is sometimes made explicit by laying out genes of the same layer in a linear fashion [38], as shown in Fig. 2.5. This layout makes it easier to distinguish regulators and to identify overlapping regulons. However, these visualizations still have many link intersections that complicate analysis of local structure. Other layout strategies, such as concentric circles [10], or planes in 3D space [39], also emphasize layers. Other GRN-specific tools are usually network editors, which use manual layouts [40].

Visualization challenges and opportunities The scale-free structure of GRNs makes them strongly interconnected and therefore hard to visualize with node-link diagrams. We have to account for the most important network tasks if we are to devise more effective visualizations: following paths, detecting local structures of interest, and inspecting neighborhoods.

Following paths and the inspection of structures at a local level is inhibited in node-link diagrams by visual clutter of the network remainder. For example, it is hard to

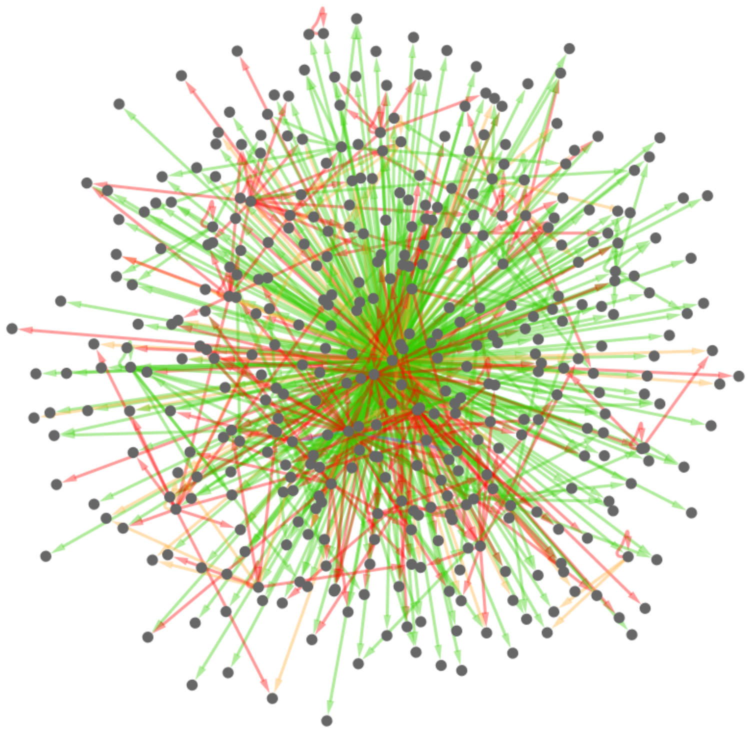


Figure 2.4: *Node-link diagram of a GRN that consists of 399 genes and 789 regulations (inhibition in red and promotion in green) laid out with force simulation in Cytoscape.*

detect a feed-forward loop in the GRN of Fig. 2.4, because its nodes are spread out and its links are obfuscated. Cutting up the network into sections of interest, similar to pathways in metabolic networks, is a viable approach. However, a definition of what is interesting is required, in addition to algorithms that derive such subnetworks accordingly. Motifs and their inference [35] are a good start, but are just instances of what may be of interest. The correspondence between established metabolic pathways and the encompassing metabolic network may provide clues for the automated derivation of GRN pathways from the global GRN topology. Nonetheless, the isolation of subnetworks has drawbacks that need to be addressed and are discussed for metabolic pathways in Section 2.3.3.

Reducing clutter in a node-link diagram while retaining global structural characteristics is possible via node duplication [44, 45]. This is a process in which clutter is reduced by representing a single gene as multiple nodes in a node-link diagram such that links do not have to converge to a single point. However, node duplication detriments the ability of an observer to follow paths and inspect neighborhoods of those

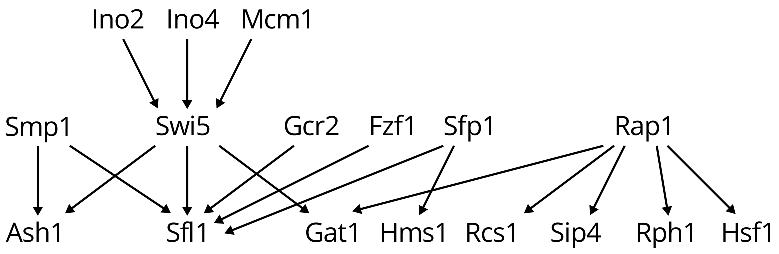


Figure 2.5: *Node-link diagram of a small GRN, laid out in layers without regulation types. Illustration derived from Schreiber* et al. *[41].*

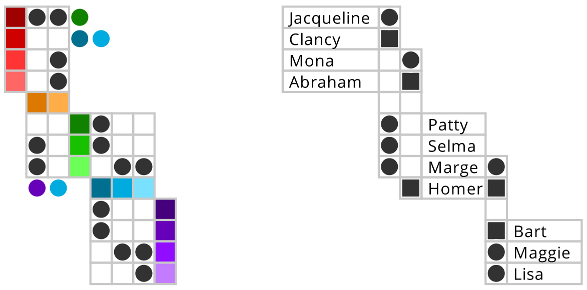


Figure 2.6: *Quilts (left) and GeneaQuilts (right), which are cascaded adjacency matrices where links that run between consecutive layers are encoded as standard markers. Links that skip layers are color coded for Quilts and extended outwards for GeneaQuilts. Illustrations derived from Bezeriano* et al. *[42] and Bae and Watson [43].*

nodes that have been duplicated.

The inspection of neighborhoods is difficult in standard node-link diagrams. The neighborhoods of smaller hubs are localized and therefore easy to interpret. However, large hubs have spread out neighborhoods that do not appear as isolated visual units. It is therefore difficult to analyze overlapping regulons as well. Visualization methods have to be developed where regulon comparison is easy and prone to little bias. Adjacency matrices, in combination with rearrangement clustering, are promising but use space inefficiently and do not adequately enable path following. These problems may be resolved by more elaborate encodings that use the layering of the GRN, such as Quilts [43] and GeneaQuilts [42] (see Fig. 2.6). Direct application of these techniques is complicated by the scale-free structure of GRNs, where many links skip several layers, causing the visualizations to become space inefficient. In Chapter 3 we therefore present a technique that leverages the same GRN layering but which results in matrix encodings that are more space efficient, and benefit neighborhood comparison and inspection of motifs.

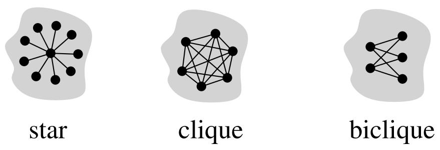


Figure 2.7: *Diagrams of important PIN subnetworks, where gray areas represent the remainder of a network.*

2.3.2. Protein interaction networks

Proteins consist of chains of amino acids that cause them to have complex dynamic geometric properties that determine a multitude of possible functions in a cell. For example, *catalytic proteins* speed up chemical reactions (see Section 2.3.3), and *structural proteins* make up the structure of a cell. Proteins are therefore an integral part of the cell itself and its chemical processes.

Proteins are in part synthesized via genes, as shown in Fig. 2.2. The presence of proteins in a cell is therefore controlled, in part, by the gene regulatory network. Proteins play an important role in gene regulation as well, as mentioned in Section 2.3.1. However, proteins also physically interact with each other. For example, two or more proteins can bind to each other to enable, disable, or combine each other's abilities in the cell. Therefore, if a protein is associated with a certain function, any (indirectly) interacting protein may also be associated and required to be present in the cell for the first protein to function properly.

The combination of a set of proteins and their interactions is called a *Protein Interaction Network* (PIN). Such networks are abstract; only the notion of interaction may be conveyed, while there are many possible ways in which proteins interact. Mapping all protein interactions for organisms is an ongoing effort, spurred on by so-called *high-throughput* techniques [46, 47]. However, some techniques are still unreliable and have low coverage [48].

Structure PINs are undirected, making the notions of cycles and layers less relevant than in GRNs. However, PINs are sparse, scale-free, and therefore also small-world [49, 50]. In particular, they are known to have dense subnetworks [51], referred to as *clusters*.

These structural characteristics imply the occurrence of several important subnetworks [52], as illustrated in Fig. 2.7. The *star* consists of a protein and its neighborhood. The frequency and size of stars match the power-law distribution of node degrees. A complete subnetwork, or *clique*, consists of a set of proteins that all interact with each other, indicating that they are able to form a larger *protein complex*. Likewise, a *biclique* consists of two sets of proteins where every protein from one set interacts with all proteins from the other set. These bicliques typically indicate the presence of proteins that have similar physical characteristcs and therefore engage in similar interactions with other proteins. These cliques make up some of the clusters present in PINs, but local clusters also result from proteins that perform more complex, dynamic functions together.

Analysis PIN analysis involves both global and local levels. Interaction connectivity on a global level indicates the importance of a protein [53]. This is known as the *centrality-lethality rule*, which states that proteins with many interactions tend to be essential to a cell (its removal is lethal). It has been shown that protein function and type are related to its position in global network structures [54]. Protein function is also predicted via *guilt by association* [55] at a local level, where the function of a protein is approximated from its interaction with other proteins.

Furthermore, cellular functions or processes are often localized to small subnetworks, termed *modules* [56, 57]. Novel modules can be discovered by combining PIN structure and differential protein expression data. Differential expression is the change in protein abundance across different cellular conditions that are induced by (artificial) changes to a cell or its environment. In Chapter 5 we present a methodology that enables the elucidation of such modules, based on pre-existing information about the proteins that comprise them.

Visualization practices Node-link diagrams laid out with force simulation are the preferred technique for visualizing PINs. Protein and interaction properties are then color coded on nodes and links respectively, like in Fig. 2.8. Certain topological features are easy to spot in such visualizations, like isolated clusters and sparse branches. However, due to the scale-free structure of PINs, there is much obfuscation and therefore no guarantee that all important information is clearly visible. Common techniques like filtering are used, which includes limiting analysis to isolated modules at a loss of context. Customized layout algorithms also exist that incorporate protein and interaction properties to generate a semantically relevant layout [58, 59].

Visualization challenges and opportunities To improve analysis of large PINs via visualization, either common node-link diagram methods have to be improved, or alternative encodings have to be developed. For example, the inspection of protein interaction neighborhoods is important for analysis. Yet, those proteins that are central to the network have large neighborhoods that are spread out and obfuscated in a node-link diagram. This sometimes makes it hard to determine whether a protein has a large (indirect) neighborhood, as is the case for the proteins at the center of Fig. 2.8. Moreover, proteins with smaller but more isolated neighborhoods draw attention, leading to biased interpretation.

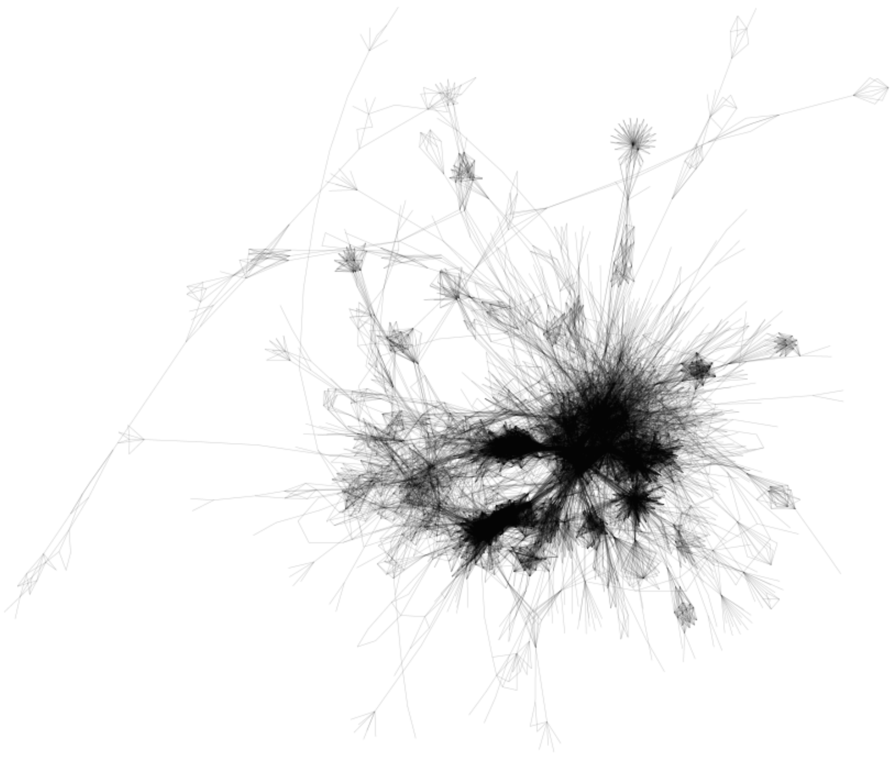


Figure 2.8: *Node-link diagram of a PIN that consists of 3852 proteins and 29293 interactions, laid out with force simulation in Cytoscape.*

The strong interconnectedness of large PINs makes improvement of node layout unlikely to have much effect when it concerns reducing clutter. More advanced link aggregation, routing [21], or bundling [22] could reduce clutter but these techniques make it harder to distinguish individual interactions while the uneven spread of neighborhoods remains.

Alternative visual encodings have already surfaced in the target domain but are not widespread. For example, it is possible to aggregate links, and therefore simplify visualizations, without loss of information. Figure 2.9 shows a Power Graph [52], where nodes are nested in a single level hierarchy. Here, a link between two internal hierarchy nodes p and q denotes that links connect from every descendant node of p to every descendant node of q. This enables the visual compression of cliques and bicliques, resulting in their emphasis, and the removal of clutter caused by many redundant links. However, the hierarchic nesting incorporates a visual bias in the encoding that may

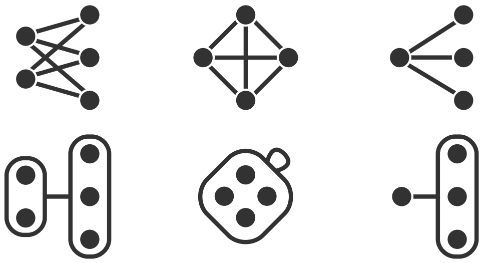


Figure 2.9: *Node-link diagrams of PIN motifs at the top and matching Power Graphs at the bottom [52].*

lead to biased interpretation.

Adjacency matrices are viable encodings as well, though hardly used in practice. Rearrangement clustering could reveal the clusters of a PIN while still conveying the global structure of the network. The overall sparseness of PINs pose a problem with respect to space efficiency of the matrix. An approach like NodeTrix [60] partially solves this by partitioning a network into dense subnetworks, conveying each as a standard adjacency matrix. The matrices are then linked together like a node-link diagram to convey the remaining links (see Fig. 2.10). Module detection algorithms are suited for such partitioning [61]. Similarly, one could provide coordinated node-link diagram and adjacency matrix views [62].

These encodings, though more structured than common node-link diagrams, still have difficulties depicting neighborhoods in a cohesive way. For power graphs, a neighborhood is composed of multiple links that can connect to nodes that are spread out over different nesting levels. Likewise, the link markers of a single neighborhood can be spread out in an adjacency matrix and NodeTrix.

However, when analysis is restricted to smaller PINs such as modules, clutter and spread out neighborhoods are minimal in node-link diagrams. Moreover, in these cases the analysis of structure, such as the comparison of interaction neighborhoods, is

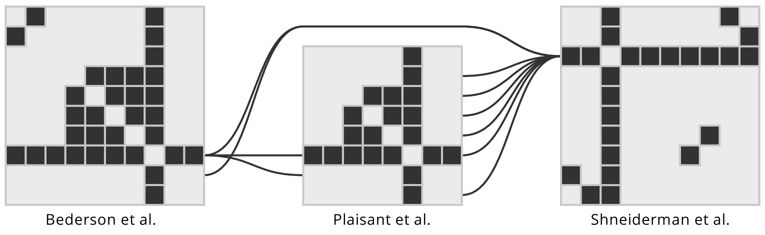


Figure 2.10: *NodeTrix representation of a citation network. Illustration derived from Henry* et al. *[60].*

of little concern. Instead, the observation of individual proteins and their interactions, in context of additional annotation and omics data, has the priority. We introduce an analysis methodology and tool for such a scenario in Chapter 5.

2.3.3. Metabolic networks

Biological cells are chemical factories that compose and decompose molecules via *reactions*, spurred on by catalytic proteins (or *enzymes*) as shown in Fig. 2.2. Every reaction converts a set of molecules, called *reactants*, into a set of different molecules, called *products*. The reactions chain together in a cell to form *metabolic pathways*. Such pathways give key insights into the functioning of a cell and are curated because of their importance. The combination of molecules and reactions present in a cell makes up a *metabolic network*, which provides insight into the entire range of possible chemical conversions in a cell.

A multitude of molecules and their reactions are known due to centuries of chemistry. However, the relatively few of these, present in a cell, are determined through measurement of compounds over time and through varying conditions in the cell. This kind of measurement has been possible for a long time and we therefore have a more complete picture of metabolism than of protein interactions and gene regulations.

Structure Metabolic networks are directed, but also bipartite with separate node types for molecules and reactions. Metabolic networks are scale-free with respect to molecule nodes, i.e., few molecules are involved in many reactions and many molecules in few reactions [63]. These networks are therefore small-world [50]. Moreover, the presence of clusters has been established as well [64].

Metabolic pathways are prominent in analysis but do not immediately emerge from the topological features of large metabolic networks. Pathways are the result of selection by priority and consensus amongst researchers, which make them important subnetworks nonetheless. Contrary to metabolic networks, the structure of pathways is simple, almost planar.

Analysis Analysis of metabolism mostly relies on inspection of already established metabolic pathways [65], which is more or less the bite sized inspection of a metabolic network at a local level. For example, researchers look at the pathways downstream from a metabolic reaction to predict the effects of its alteration (by manipulation of the proteins that catalyze the reaction). Similarly, the production of a particular metabolite could be promoted by enforcing particular reactions. However, it is possible for chains of reactions to span multiple pathways. Pathways have to be overlaid or stitched together to find such chains, creating metabolic networks of varying size and complexity. The global structure level is of interest, as well as finding modules [66] that are responsible for specific functions in the cell and are preserved across species of an

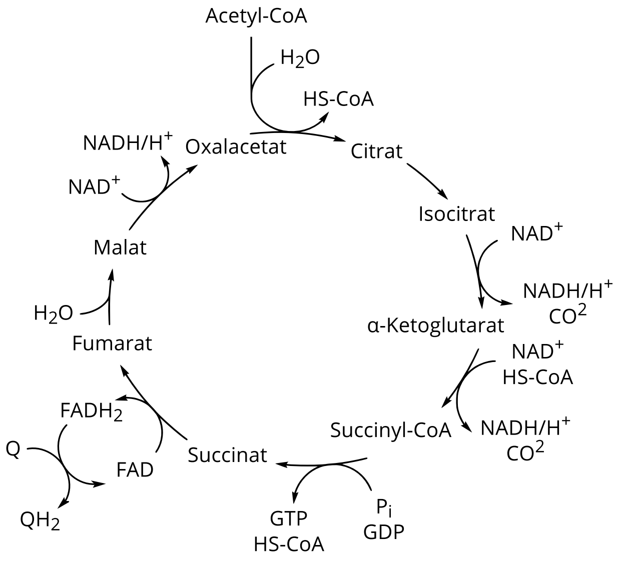


Figure 2.11: *Typical poster-style drawing of the citric acid cycle.*

organism. For example, the elements of a module are often strongly interconnected but are less interconnected with the remainder of the network.

Visualization practices Traditionally, metabolic networks and pathways are drawn manually and presented on posters [1]. The drawing style is the same as in standard textbooks, and online databases such as BioCyc [67] have adopted this style as well. An example can be seen in Fig. 2.11, which shows part of the citric acid cycle. In these networks, nodes represent compounds and directed links encode their transformations. Pathway images in KEGG [68] are also constructed manually, but now using the bipartite graph style. See Fig. 2.12 for an example of the human vitamin B6 metabolism in KEGG. In this image, a compound is represented as a circular node and an enzyme as a rectangular one. Links to other pathways are made explicit by drawing rectangles with rounded corners.

However, manually laying out networks is time consuming when used in an analytic context, where the network topology is dynamic due to user interaction. This creates the need for automated layout algorithms. Standard network layout algorithms, such as circular layouts, hierarchical layouts, and force-directed layouts, turned out to be unsatisfactory. The layouts produced do not reflect textbook standards, which makes the visualization ill-suited for domain experts. To overcome this problem, layout algorithms that are specific to pathways have been proposed which mimic textbook drawing styles [70-72]. In some cases, it is important to reflect the cellular locations of reactions in the layout to understand the biology underlying the pathway. Recently,

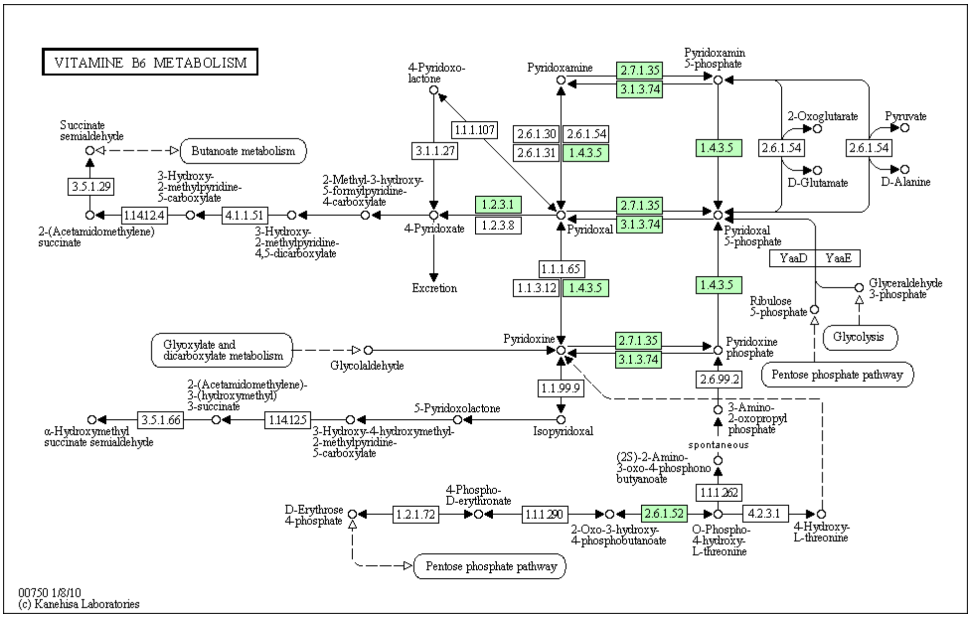


Figure 2.12: *Vitamin B6 metabolism in humans. Image generated with KEGG [68].*

Kojima *et al.* proposed an approach that takes this into account [69]. An example of their method can be seen in Fig. 2.13, which shows the plasma membrane, cytoplasm, mitochondrion, and nucleus as concentric boxes; the nodes of the network are placed in one of these layers.

Drawing a full metabolic network by a force-directed algorithm is not very useful for a biologist. Figure 2.14 shows the metabolic network of *E. coli* laid out by the FM^3 algorithm [73]. Clearly, particular topological features, such as cycles and cascades, are not distinguishable, and pathway information is not retained either.

Bourqui *et al.* [76] proposed a layout algorithm to handle these issues. The method

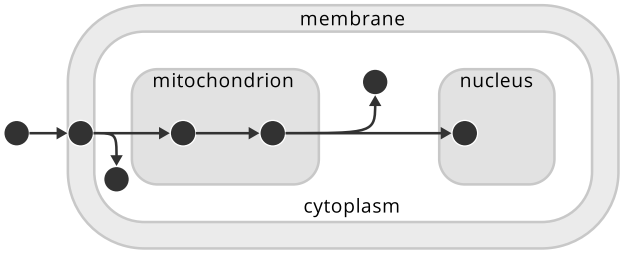


Figure 2.13: *Location-aware layout algorithm, which takes the cellular location of reactions into account. Illustration derived from Kojima* et al. *[69].*

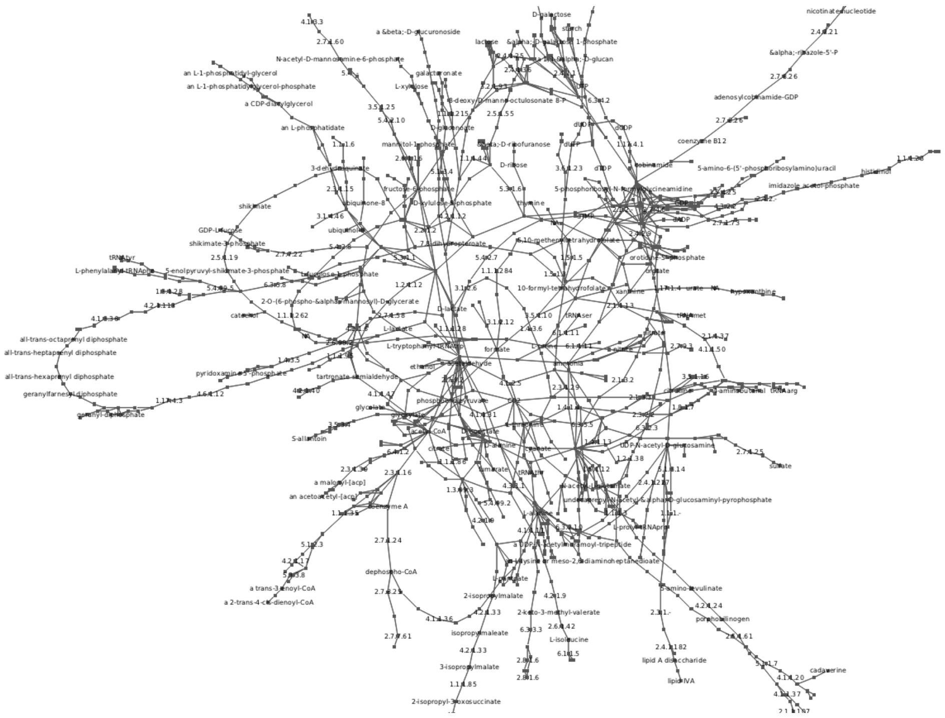


Figure 2.14: *Metabolic network of* E. coli *[74] drawn with a force-directed layout algorithm [75].*

starts by computing maximal sets (clusters) of independent pathways. Two pathways are independent if they do not share any enzyme or compound. Next, topological structures are detected in each cluster, and laid out accordingly: cycles on circles, cascades on straight lines, and the hierarchical organization of pathways is emphasized using a hierarchical, or layered, layout. The clusters themselves are represented and visualized by metanodes, and all links between two of these clusters are replaced by a single metalink. An example is shown in Fig. 2.15 where metanodes are displayed as either blue (complete pathway), light blue (partial pathway), or yellow boxes (topological structure).

In KGML-ED, a different approach is taken [77]. This method arranges KEGG pathway maps in an overview network, where a node represents a pathway and links between nodes represent connections between pathways. The layout of the overview network can be computed by any suitable layout algorithm while the layout of each pathway is exactly as specified by KEGG. This ensures that pathway drawings conform to expected standards. Figure 2.16 shows an example of a small overview network containing five pathways and the expanded network containing pathways and

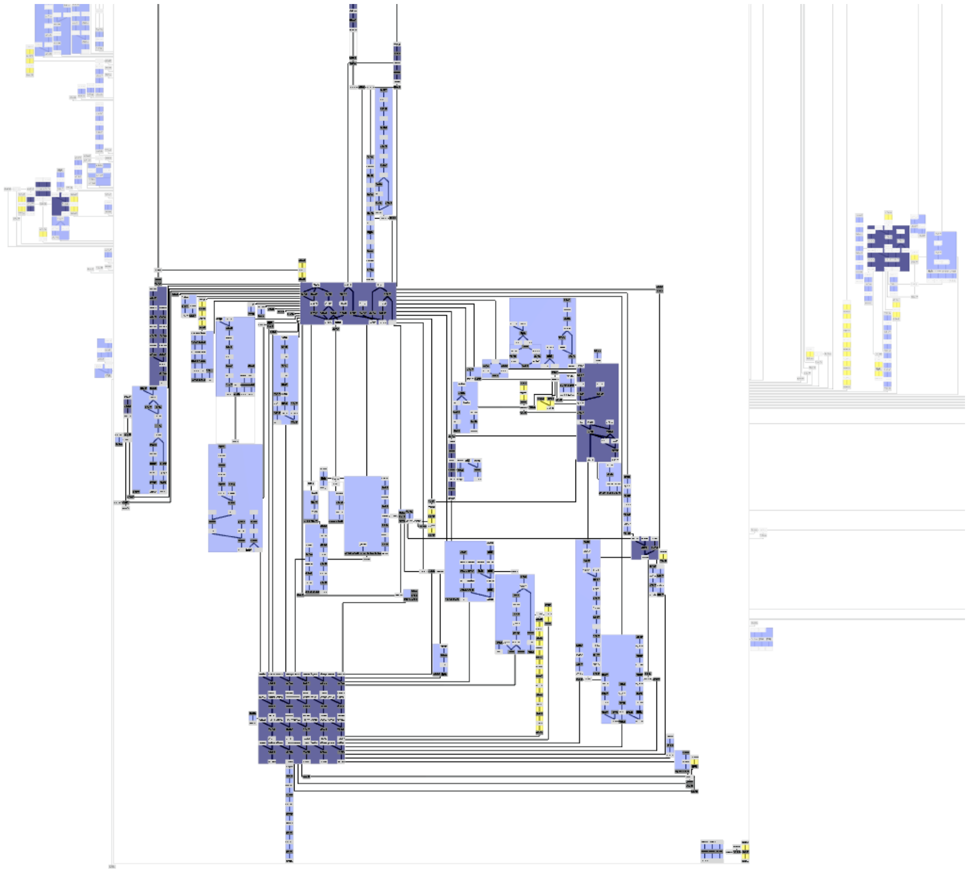


Figure 2.15: *Metabolic network of* E. coli *[74], drawn with the MetaViz layout algorithm [76].*

connections between two nodes.

Visualization challenges and opportunities The proper visual concatenation of already laid out metabolic pathways is a major concern and has been voiced before [65, 78]. Some of the discussed techniques address this, but improvement is still possible. Individual pathway layouts have to be retained as much as possible to preserve the mental map of analysts. However, following paths across many concatened pathways is difficult, as shown in Fig. 2.15 and 2.16.

Overlaying pathways is an alternative approach to cutting up pathways and placing them beside each other. By manipulating their configuration in a constrained manner, it would be possible both to discern individual pathways and to follow more complex paths. In addition, a more interactive pathway-centered exploration methodology

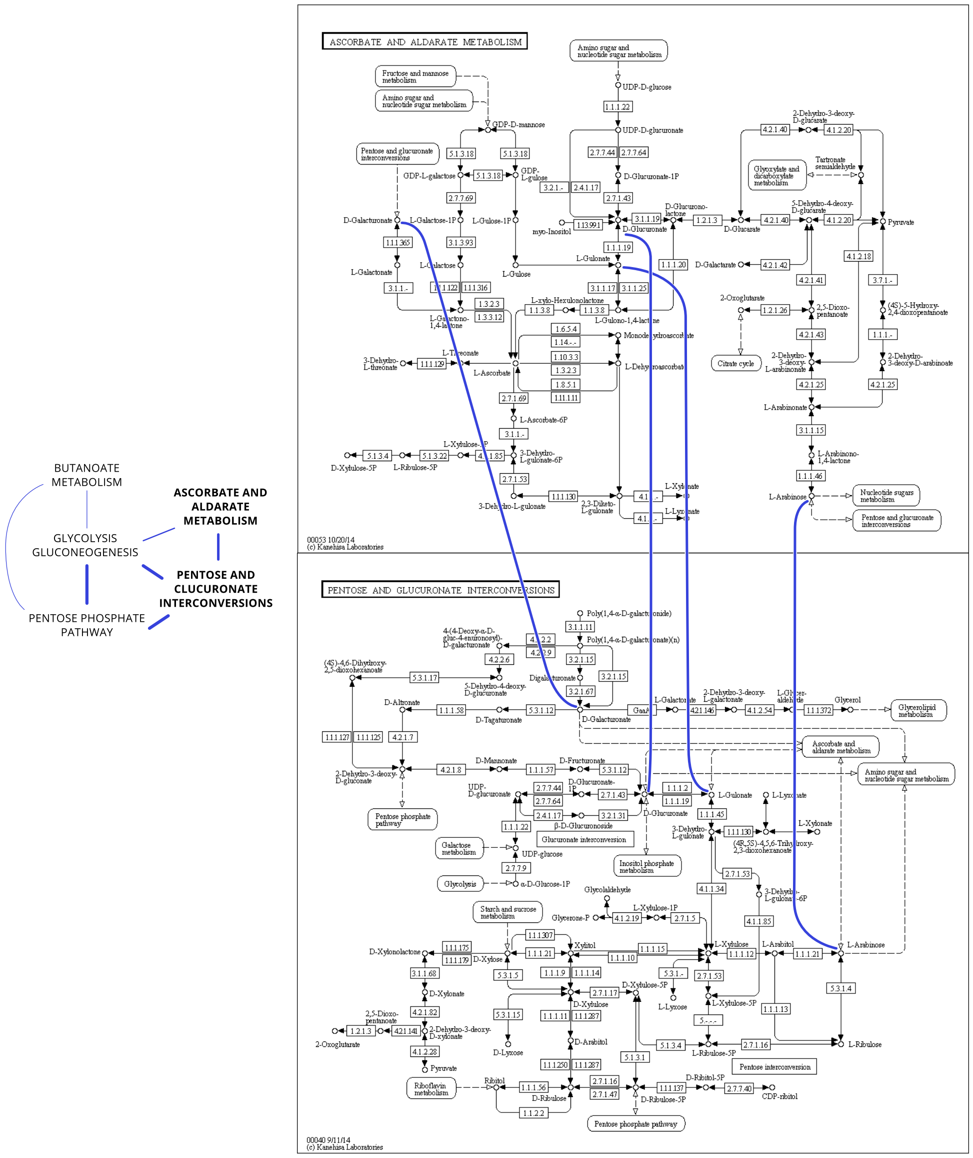

BUTANOATE METABOLISM
ASCORBATE AND ALDARATE METABOLISM
GLYCOLYSIS GLUCONEOGENESIS
PENTOSE AND CLUCURONATE INTERCONVERSIONS
PENTOSE PHOSPHATE PATHWAY
ASCORBATE AND ALDARATE METABOLISM
00053 10/20/14
(c) Kanehisa Laboratories
PENTOSE AND GLUCURONATE INTERCONVERSIONS
00040 9/11/14
(c) Kanehisa Laboratories

Figure 2.16: *KGML-ED visualization style. Overview network of KEGG pathways (left) and two expanded nodes displaying actual pathways and interconnections (right). Images derived from Klukas* et al. *[77].*

should be considered. Then, fewer joint pathways have to be shown at a single time, and pathways can be dynamically added and removed. This way, the entire metabolic network can be explored while the analyst's visual orientation is preserved via smooth animation of changes in joint pathway configurations. Computing such non-invasive configuration changes is a challenging problem but could be derived from existing techniques used for dynamic graph drawing [79]. These techniques could then also be used to deal with the mapping of heterogeneous data on pathways in an interactive context, which requires a lot of space and places additional constraints on the layout of pathways.

2.4. Network Integration

Most work has focused on studying gene regulatory networks, protein interaction networks, and metabolic networks separately. The aim of systems biology [34, 80-82] is to combine and integrate these individual networks to a *network of networks*, in order to provide insight into the overall behavior of a cell. For example, to understand and interpret biological function, it is helpful to explicitly show feedback control of metabolites on regulator activity. In network terms, this requires a link between a metabolite (a node in the metabolic network) and a regulator gene (a node in the gene regulatory network).

Integration of networks is not simply a matter of combining them into a larger network with additional node and link types. There are many challenges to overcome when integrating various data sources [84, 85] and computational modeling [86-88].

Visualization practices Rather than explicitly constructing joint networks, current visualization approaches aim to visually integrate the networks or parts thereof. There are essentially two approaches: (1) overlay additional information over an existing visualization of a network [83] and (2) make use of coordinated and multiple views [89].

An example of the first approach is shown in Fig. 2.17. The visualization combines a pathway image and a protein interaction network. The colored nodes are all part of the pathway and most of the protein interaction network is visible in the background. The protein with label *38* is selected and its interactions are highlighted by blue links. Here, a potential problem is the limited space for drawing the additional links without occluding the underlying pathway image.

In the second approach, each network is visualized in a separate view, and these views are synchronized in some form. For example, nodes selected in one view are highlighted in the other view(s) to indicate that the items are the same or related to each other. An example can be seen in Fig. 2.18, which shows some highlighted nodes in parts of a gene regulatory network and a metabolic pathway. All prominent biological network visualization tools provide this simple functionality of highlighting. However, from a perceptual point of view, highlighting suffers from two problems: (i) if the vi-

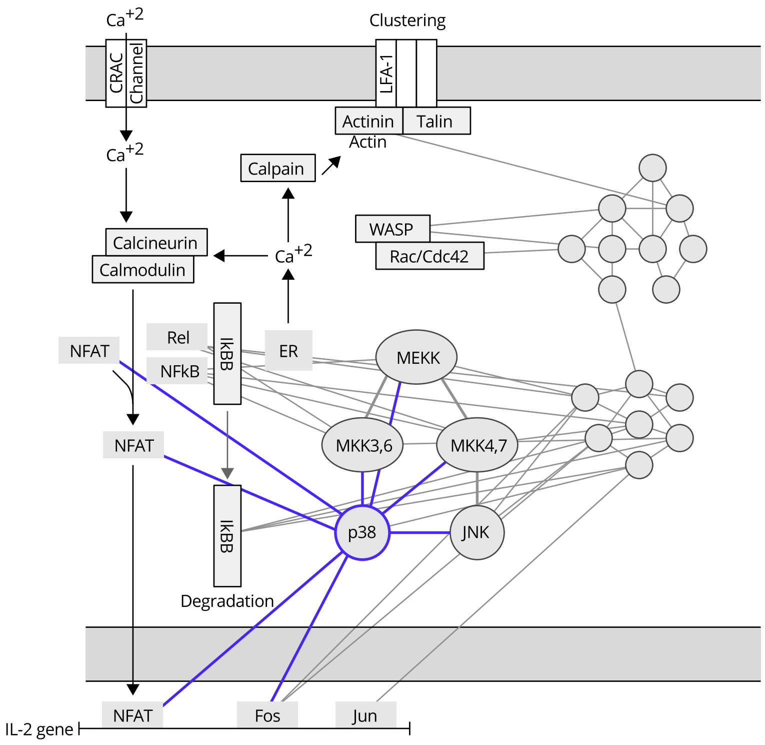


Figure 2.17: *Visualization of protein interactions integrated into a pathway image. One protein is selected and its interactions are highlighted (blue links). Image derived from Jianu* et al. *[83].*

sualization itself makes use of different colors, it is difficult to choose a highlighting color that stands out from the other colors, and (ii) thin and small frames drawn around highlighted items are difficult to perceive.

Visualization challenges and opportunities To overcome the problems of simple highlighting, a technique called *visual linking* has been proposed in the visualization community [90-92]. This technique connects highlighted items with (colored) lines, curves, or surfaces to make the relations explicit. As this may cause visual clutter and occlusion of items of interest, a more sophisticated form of visual linking, called *context-preserving* visual links [93], has been proposed recently. This approach takes the underlying visualization into account and routes the links around important parts to minimize occlusion. In Chapter 4 we introduce a similar technique, which is also demonstrated for a metabolic pathway.

Drawing *joint* networks in an understandable way is a challenging problem. Since

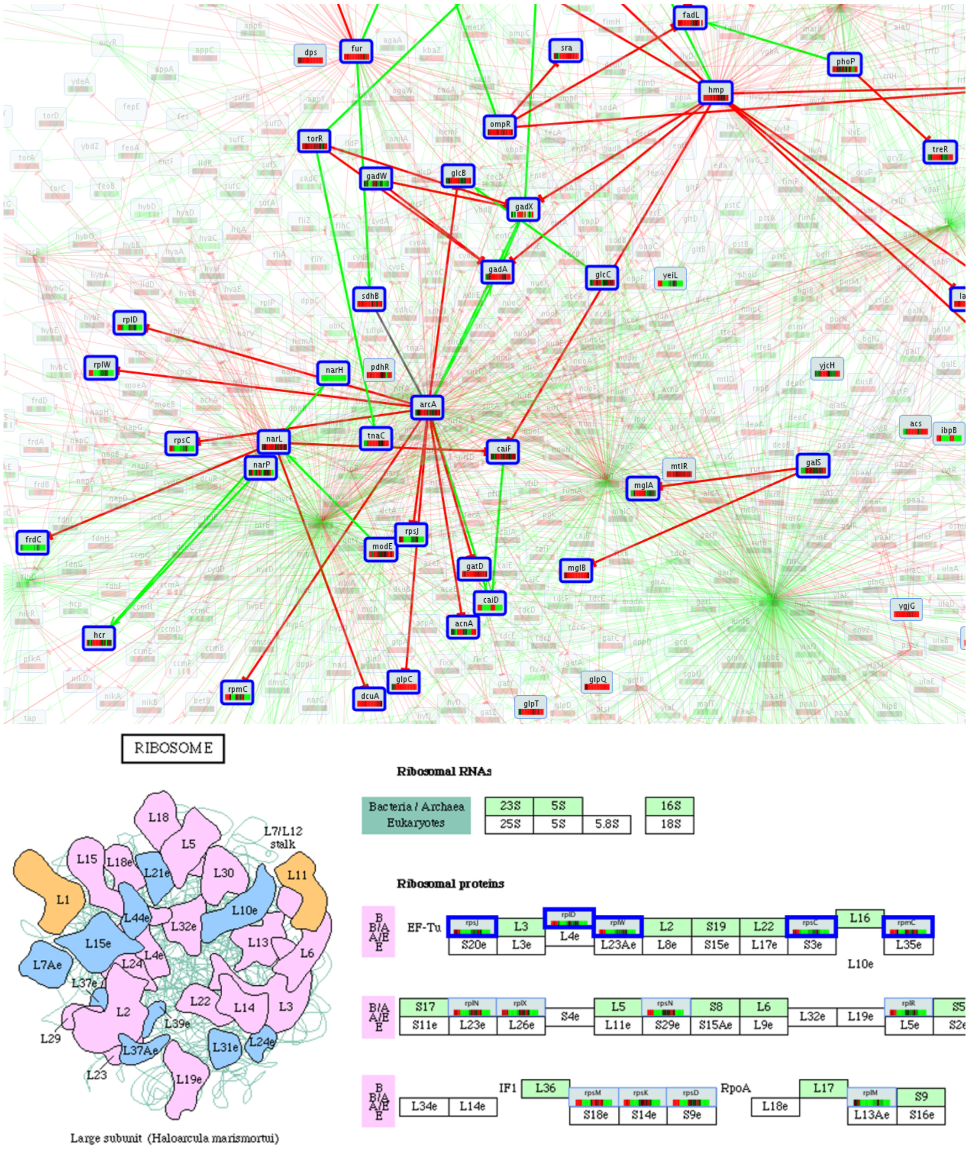


Figure 2.18: *Coordinated views of a gene regulatory network (top) and metabolic pathway (bottom). Selected nodes are highlighted by a blue border. Images generated with GENeVis [15].*

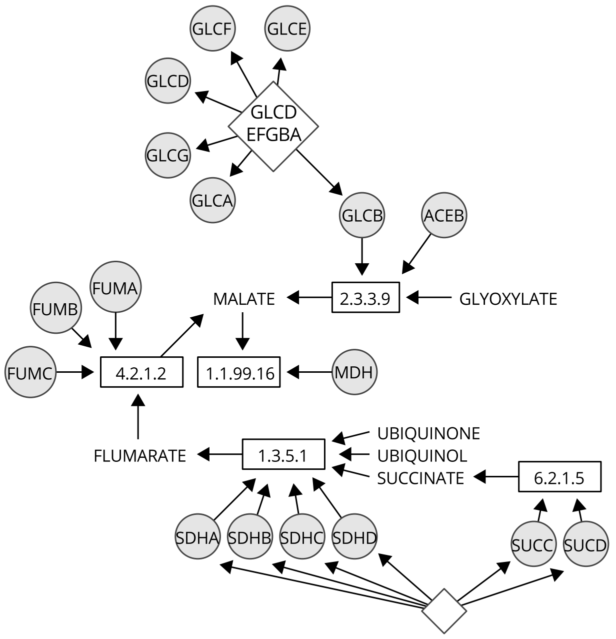


Figure 2.19: *Visualization of a joint metabolic and gene regulatory network, derived from Yeang* et al. *[86].*

it is unclear what the structural properties of such networks are, it will be hard to design specialized and effective layout algorithms. Also, there are different conventions for drawing the individual networks, which raises the question whether it is even feasible to combine these into one view. For example, Fig. 2.19 shows a part of a combined gene regulatory and metabolic network from a joint model [86]. This visualization is quite difficult to read because there are different node and link types. Furthermore, the size and color of nodes and links can have various meanings. Finally, the layout of the metabolic part of the network does not resemble the standard textbook style.

The Systems Biology Graphical Notation (SBGN) [94] may help to overcome the problems with graphical notation of different node, link, and network types. The SBGN is a visual language, similar to the Unified Modeling Language (UML) for software engineering. However, in the same way that UML is not a software visualization approach, SBGN is not a biological network visualization approach. While standardization of models and notation is an important step, it is clear that major visualization challenges are still to be tackled.

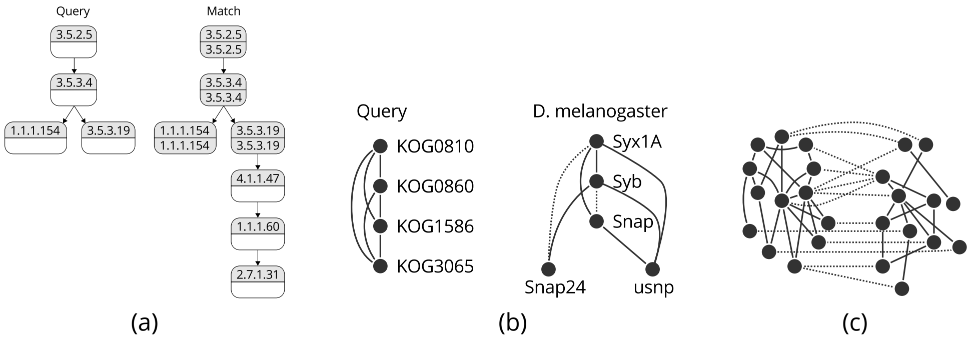


Figure 2.20: *Network alignment visualization approaches: (a) Combined nodes, image derived from Pinter* et al. *[100]. (b) Nodes on the same horizontal line, image derived from Koyutürk* et al. *[101]. (c) Nodes in approximately the same relative position, image derived from Sharan* et al. *[102].*

2.5. Network Alignment

The main use of network alignment in cell biology is the detection of subnetworks that are conserved across species. Here, a particular subnetwork with known functions is extracted from one network, and a subnetwork with an (almost) identical structure is determined in another network. This matching of networks enables biologists to locate proteins that have similar functions across different species. It also enables them to find functional modules in one species, based on known functional modules in other species. It is expected that network alignment techniques will take a leading role in bioinformatics research [95].

While sequence alignment has a long history in biology, network alignment is still in its infancy and not free of methodological problems [48]. In general, the problems of network comparison and alignment is intractable since they can be reduced to the subgraph isomorphism problem. Finding isomorphic subgraphs is known to be NP hard. In most cases, methods are used that address this problem heuristically [96-98]. However, some recent work shows that it is possible to globally align protein interaction networks without resorting to a heuristic approach [99].

Visualization practices In a review paper, Brasch *et al.* [103] presented an overview of the main approaches for visual network comparison. We briefly summarize their main findings first and then describe some more approaches in the following paragraphs. Approaches so far either employ side-by-side views or construction of a single new network. Side-by-side views show networks next to each other, and aligned nodes are connected by links. To avoid clutter, aligned nodes can be drawn on the same horizontal line [101] (see Fig. 2.20(b)) or at roughly the same relative position in

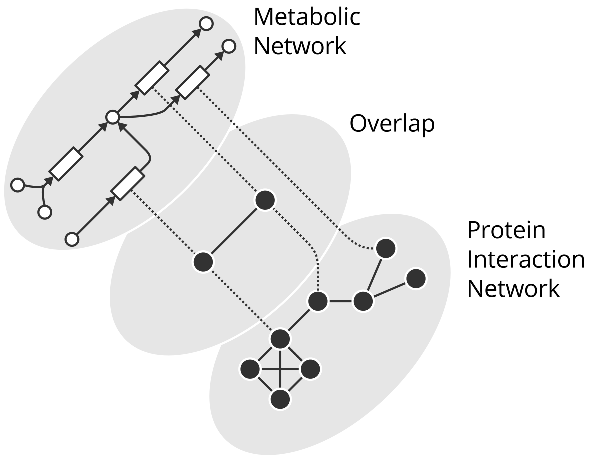


Figure 2.21: *Visualization of overlap between a metabolic network and a part of a protein interaction network. The overlap between proteins (enzymes) and adjacent reactions is shown in a center plane. Image derived from Fung et al. [104].*

each network [102] (see Fig. 2.20(c)). Either approach may produce satisfactory results, if few small networks are compared simultaneously. Aligned nodes can also be combined into a new node, resulting in a new network [100] (see Fig. 2.20(a)). In this case, it is not clear how to handle the links of the original networks. The links can be maintained, which can lead to many links between any two nodes, or they can be combined into a single new link. In the latter case, the information in the network in which the link originally occurred is lost. A further complicating factor in visualizing aligned networks is posed by metabolic networks, where domain experts impose constraints on the layout of the individual networks as we have seen before.

For the visualization of overlap between two networks, 2.5D solutions have been explored [104]. The idea is to use three planes embedded in a 3D space: one plane is used to draw each network, and the third plane explicitly represents the overlapping part. Figure 2.21 shows an example of this technique. This approach works reasonably well for small networks as shown in the picture. However, for large networks, it becomes a challenge to see both the individual networks and the overlap clearly. The (potentially) many links result in a cluttered display, the individual planes will overlap in the projection to 2D, and a lot of user interaction will be necessary to understand the visualization. This approach also does not allow comparison of more than two networks.

Another 2.5D approach [105] is shown in Fig. 2.22. Here, each network is embedded on a 2D layer, with the constraint that aligned nodes are kept at the same position. Stacking the layers in 3D space effectively results in a 2.5D visualization. For small

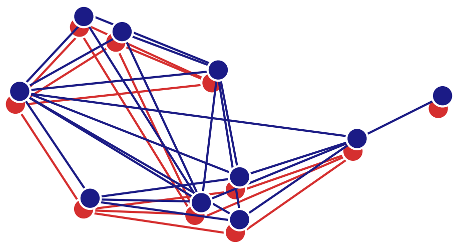


Figure 2.22: *Visualization of two aligned networks in 2.5D space. Image derived from Brasch et al. [105].*

networks, this approach works reasonably well, as can be seen in Fig. 2.22. However, for larger networks, this approach breaks down because of too much clutter.

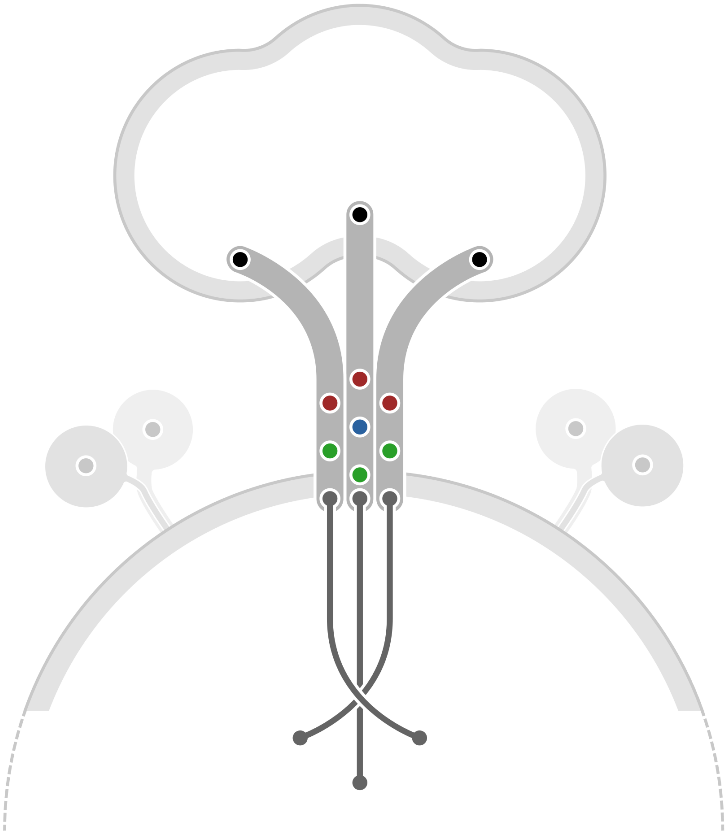
Visualization challenges and opportunities The visualization of aligned networks is a challenging problem for which hardly no good solutions exist. The difficulties arise from the fact that both individual networks and the overlap between them needs to be clearly visible and understandable. For small networks, most of the current approaches are usable. However, they all fail on larger networks and complex overlaps.

The visualization community has worked extensively on comparison of trees and hierarchies [6]. However, comparison of networks has received little attention. Even in recent work, comparison is restricted to two networks [106]. This approach does not preserve the layout of the compared networks either, which makes it unsuitable for application to the biological domain. Visual comparison of multiple networks is still in its infancy, and there are many problems for which no obvious solutions yet exist.

2.6. Conclusion

We have discussed the difficulty of visualizing specific types of biological networks, including their integration and comparison. These biological networks and associated visualization problems will reappear in the subsequent chapters. However, there are problems beyond visualization of network structure that have to be solved as well and are poised to be important in the future [78]. For example, there is a need to visualize elaborate attributes attached to nodes and links, including nominal and time-series data. This data adds a new dimension to the problem of network layout because biologists want to analyze the data in network context, requiring ample space close to nodes and links to visualize it. This requires the development of new kinds of visualizations. A good example of such an approach is Pathline [107], where metabolic pathways are flattened to one dimension to ease the analysis of attribute data while maintaining a biological context.

3
Compressed Adjacency Matrix



K. Dinkla, M.A. Westenberg, and J.J. van Wijk

Parts of this chapter have been published in
IEEE Transactions on Visualization and Computer Graphics [211].

We present a novel technique, Compressed Adjacency Matrices, for visualizing gene regulatory networks. These directed networks have strong structural characteristics: out-degrees with a scale-free distribution, in-degrees bound by a low maximum, and few and small cycles. Standard visualization techniques, such as node-link diagrams and adjacency matrices, are impeded by these network characteristics. The scale-free distribution of out-degrees causes a high number of intersecting edges in node-link diagrams. Adjacency matrices become space-inefficient due to the low in-degrees and the resulting sparse network. Compressed adjacency matrices, however, exploit these structural characteristics. By cutting open and rearranging an adjacency matrix, we achieve a compact and neatly-arranged visualization. Compressed adjacency matrices allow for easy detection of subnetworks with a specific structure, so-called motifs, which provide important knowledge about gene regulatory networks to domain experts. We summarize motifs commonly referred to in the literature, and relate these to network analysis tasks common to the visualization domain. We show that a user can easily find the important motifs in compressed adjacency matrices, and that this is hard in standard adjacency matrix and node-link diagrams. We also demonstrate that interaction techniques for standard adjacency matrices can be used for our compressed variant. These techniques include rearrangement clustering, highlighting, and filtering.

3.1. Introduction

Standard network visualization methods, the node-link diagram and adjacency matrix, are popular due to their intuitive simplicity and their generic applicability. However, they do not scale well for complex networks, leading to the infamous hairballs in case of node-link diagrams. Recent developments in the field of network visualization move towards the exploitation of predetermined network characteristics in order to create visualizations that are more effective for analysis by domain experts. We present such an approach, which has been specifically designed for gene regulatory networks of bacteria.

The gene regulatory network (GRN) of a bacterium describes interactions between a gene product (a regulator protein) and its target genes. Genes are specific pieces of DNA that form the blueprint for the creation of proteins. A regulator gene, a gene that codes for a regulator protein, can either promote (further) or inhibit (impede) other genes. Sometimes, a gene both inhibits and promotes, depending on environmental conditions. This regulation (or control) between genes is described by a GRN, where genes and regulations are represented as nodes and edges, respectively.

Because GRNs describe part of a bacterium's internal mechanics, domain experts study this type of network intensively, by making use of standard network visualization and layout methods. However, the specific structural properties of GRNs make visualization difficult, even when the network is small. This difficulty is caused by the following characteristics [34]:

Low in-degree Every gene is regulated by only a few other genes. Therefore, all nodes of the network have a low in-degree.

Scale-free out-degree There are few genes that regulate many others, and many genes that regulate few others. Therefore, the network's out-degree distribution follows a power law.

Few cycles The network has few cycles because genes rarely (indirectly) regulate each other both ways.

In this chapter, we propose an approach that exploits these structural characteristics, by creating an adjacency matrix that is cut up and rearranged to take up less space. Our novel network depiction benefits bacterial GRN analysis, and has the following properties:

Compactness Enables a detailed overview of the entire network;

Localization of motifs Enables quick detection of subnetworks of interest;

Consistent arrangement Enables interaction while preserving visual context.

3.2. Related Work

Much research in the area of network visualization has been performed [5, 6]. In this section, we restrict our overview to related work directly relevant to our proposed approach.

The node-link diagram and adjacency matrix are the most popular for network visualization. Node-link diagrams are, in general, used to depict sparse networks (with a low edge to node ratio), while adjacency matrices are an alternative for dense networks (with a high edge to node ratio) [108]. Node-link diagrams are intuitive, and most people are able to interpret such diagrams without receiving further explanation. Therefore, network visualization tools in the biological domain often use this representation [2, 11, 13, 15]. However, when a network is dense, this intuitiveness is overshadowed by the visual clutter caused by the many edges that have to be drawn. In this case, an adjacency matrix provides a more ordered and therefore easier to interpret depiction, even though an observer will have to become accustomed to the visualization [109, 110].

Optimal drawing of node-link diagrams, corresponding to aesthetic criteria, and the combinatorial aspects of deriving diagrams that adhere to these criteria, is the focus of the *Graph Drawing* domain [16]. Eades was the first to propose a force simulation to lay out graphs [18], after which more elaborate force models were developed [19].

Such force-based layouts are simple and effective. Aesthetic criteria are used to judge the effectiveness of node-link diagrams and to derive algorithms that generate high-quality diagrams [16]. The number of intersections in a diagram is an important criterion, and therefore many techniques focus on reducing these intersections [111-113]. Giving edges a more complex geometry than a straight line, enabling the routing of edges, is a way to avoid intersections as well [21].

A simple example of the exploitation of specific network characteristics is that of tree-layout algorithms [114]. Such algorithms generate layouts that emphasize the inherent tree structure. Similarly, DAGs (short for Directed Acyclic Graphs) are given special treatment with Sugiyama's algorithm [38], where the layered structure of a DAG is exploited and emphasized.

The layout of an adjacency matrix is fairly rigid, i.e., nodes and edges are restricted to rows and columns. Most attention has been given to ordering nodes in such a way that patterns in the network structure become apparent. This is known as *rearrangement clustering*, where nodes are placed close to each other in the arrangement when their distance, according to some similarity measure, is small as well [23]. Such approaches take on more complex forms in bi-clustering [25], which tries to optimize horizontal and vertical ordering of nodes simultaneously. The use of adjacency matrices and corresponding rearrangement techniques is popular in the biological domain, where they appear in the form of *heat maps* [24].

Many analysis tasks involve following a path along the edges of a network. In node-link diagrams, edges can be traced easily from node to node, provided they are not overly obfuscated by intersections with other edges. Node duplication involves the placement of multiple copies of the same node, and distributing the edges over these duplicates [44]. This enables greater reduction of edge crossings, creating a cleaner depiction, at the cost of paths becoming harder to trace over the nodes that have been duplicated. This problem also occurs in adjacency matrices, because every node is represented horizontally as well as vertically, for outbound and inbound edges, respectively. Therefore, following a path in an adjacency matrix is notoriously hard [108].

In a biological setting, node duplication is often used for node-link diagrams of metabolic networks [115, 116], where certain nodes have such a high degree that they are either split into many duplicate representations or removed altogether. This is done because some compounds, like water, are involved in so many metabolic processes that they do not represent information of particular interest to an analyst. The tasks that an analyst has to perform, aided by a network visualization, are of great influence to such trade-offs. In some cases, tracing paths is of no concern and representations more akin to adjacency lists are used, such as animal pedigrees [117]. Here, paths are hard to follow, but it is much easier to compare two nodes (and their attributes) that share an edge. Dealing with the richness of information attached to nodes and edges of networks often leads to these kind of solutions in visual analytics. *Pathline* is another example of this [107], where metabolic pathways are shaped into a vertical linear ordering, making it harder to interpret the pathway, but easier to compare corresponding time-series data that can now be stacked on top of each other in a consistent manner.

Any network visualization is prone to scalability issues. Large graph layout algorithms [118] often result in visualizations that are difficult to interpret. In such circumstances, compression of the network, either lossless or lossy, can be beneficial. Here, lossless compression refers to a visualization that still encodes all network information, but with more complex encoding schemes that require additional effort by an analyst to interpret. For example, combining nodes with edges identical to other nodes (identical neighborhoods) in a joint node can drastically reduce the number of edges that have to be drawn. However, the routing of edges to nested nodes makes the structure of nested nodes more complex. Grouping together children of trees is a simple version of this [119], and an extension of this encoding to multiple levels of nesting has been applied for the analysis of protein interaction networks, forming so-called *Power Graphs* [52]. Lossy compression is usually done by clustering of nodes into meta-nodes, thereby also collapsing edges into meta-edges, where the presence of individual nodes and edges may be lost in the visualization [120]. These kind of approaches are used in interactive settings, where meta-nodes may be expanded and collapsed at will [121].

3.3. Motivation and Concept

Node-link diagrams are the customary method of inspecting gene regulatory networks in the biological domain, see Fig. 3.2 for an example. Global features of the network are visible in the diagram, such as the strong interconnectedness of the network, few genes that regulate many other genes, and many genes that are regulated by few genes. More detailed features, such as the immediate neighborhood of a gene, are obfuscated by many links causing substantial overlaps. Yet, context and details are both important to biologists: they require a clear overview of the immediate neighborhood of a gene, but also need to follow paths and determine indirect regulations. Our aim is to facilitate both in the same visualization.

As an alternative visualization, it would be possible to use an adjacency matrix. However, as is clear from Fig. 3.2, the network contains many nodes but relatively few edges: it is sparse. The resulting adjacency matrix would therefore take up a lot of display space (approximately 700 rows and columns), but it would be mostly empty. This makes it hard to see both network context and details at the same time. However, the specific characteristics of a GRN, low in-degree, power law out-degree, and few and small cycles [34], allow us to introduce a variant of an adjacency matrix that we call a *Compressed Adjacency Matrix* (CAM).

The first two characteristics can be seen clearly in Fig. 3.2. It is important to note that they cause a strong interconnectedness of the network, which makes it difficult

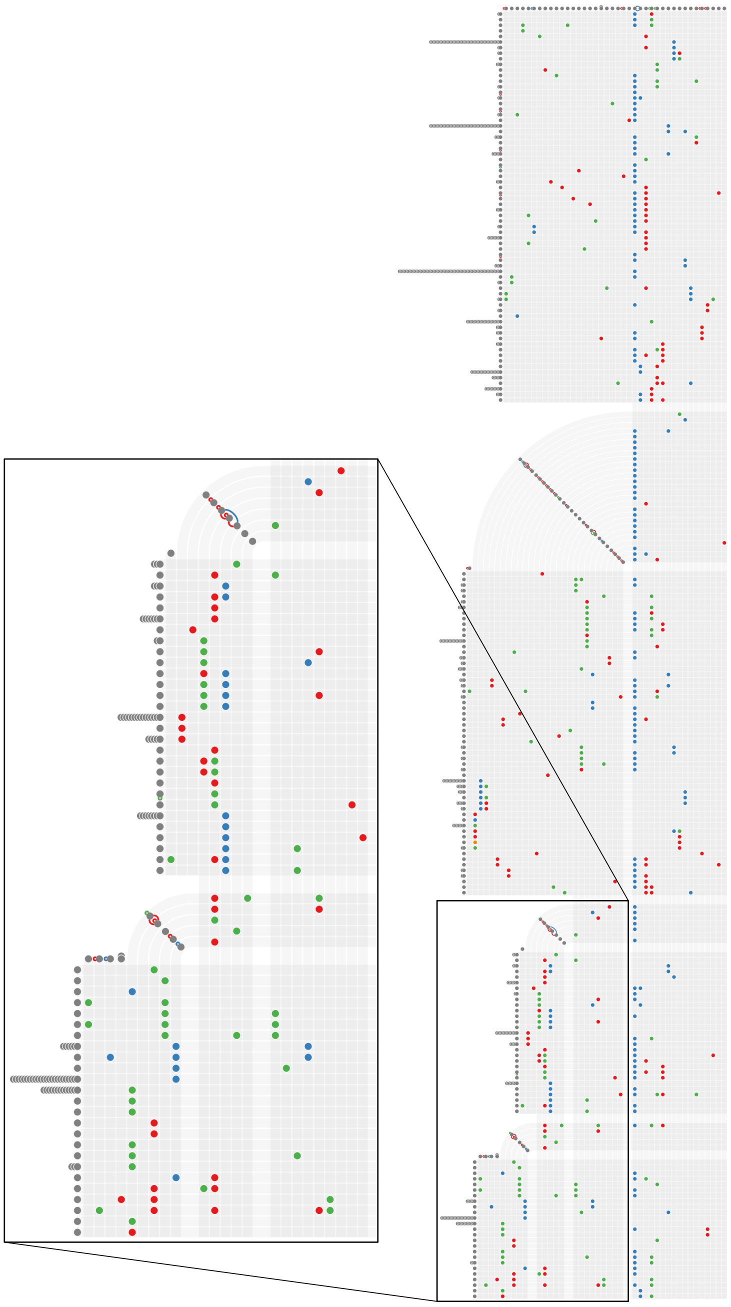


Figure 3.1: *Rotated CAM of the gene regulatory network of Bacillus subtilis, which consists of approximately 700 genes and 1000 regulations [122].*

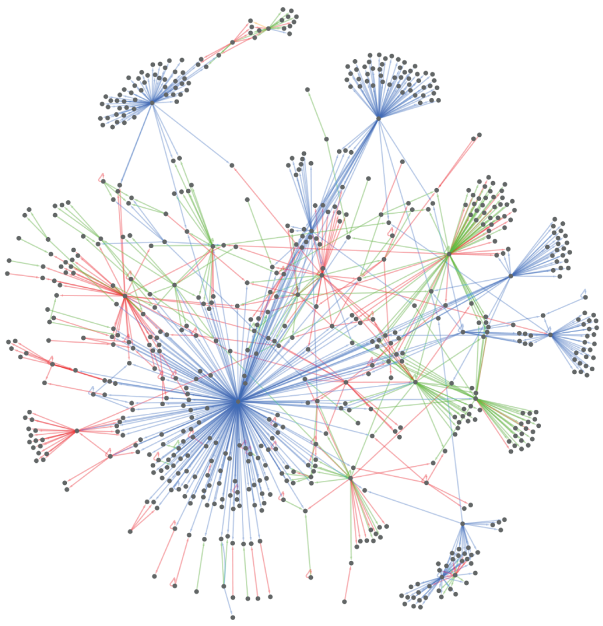


Figure 3.2: *Node-link diagram of the same GRN as in Fig. 3.1. Generated with Cytoscape [2], using a standard spring-embedder layout algorithm. Genes are depicted as dark gray nodes and regulations as colored links: green, red, orange, and blue for promotion, inhibition, both, and unspecified, respectively.*

to interpret a node-link representation. However, in combination with the third characteristic, they imply that a GRN has a DAG-like structure that consists of few layers. Moreover, because of the second characteristic, there are relatively few *roots*, nodes with no inbound edges; few *hubs*, nodes with both inbound and outbound edges; and many *leaves*, nodes with no outbound edges. This is essential to the construction of our CAM, and they are crucial to obtain its regular structure (see Fig. 3.1).

The interpretation of a CAM is straightforward, but, like a standard adjacency matrix, a CAM is somewhat less intuitive than a node-link diagram. Fig. 3.3(c) illustrates how to read a CAM: The outbound and inbound edges of a node are found by looking to the right and upwards, respectively, while using the underlying thick lines as guides.

From Fig. 3.3, several beneficial properties of CAMs become apparent. First, the entire neighborhood of a node is easy to find by following its underlying line. Second, nodes with identical neighborhoods are localized as stacks that save space (lossless compression). Third, the few and small cycles in the network are localized and give the impression of an actual cycle, due to the edges of the cycles being drawn as arcs. Fourth, the layers of the network's DAG-like structure, and regulations between these layers, are compartmentalized and visible as contiguous surfaces. Finally, paths can

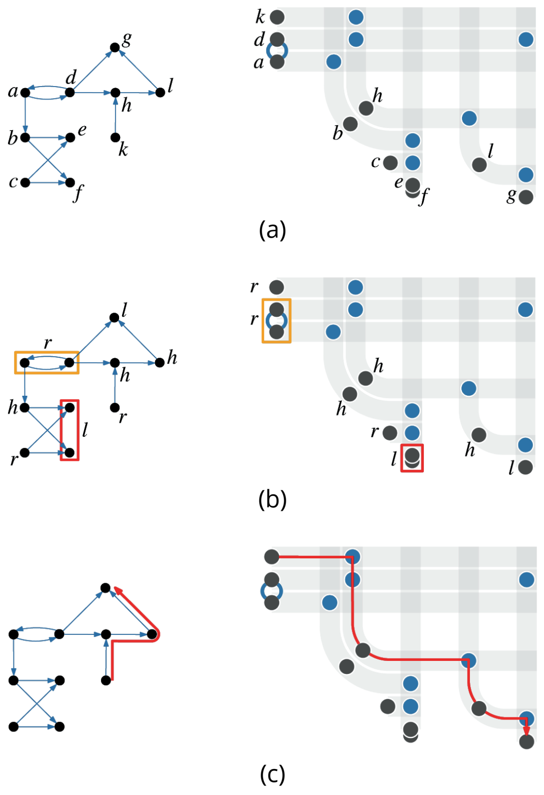


Figure 3.3: *Node-link diagrams of a simple network on the left and corresponding CAMs on the right:*
(a) Nodes are uniquely labelled and edges colored blue in both representations, the CAM shows edges as arcs or dots.
(b) Cycles are grouped in the layout of a CAM (orange), and nodes with identical neighborhoods are stacked (red), roots are placed before edges (labelled r), hubs diagonally (labelled h), and leaves at the bottom (labelled l).
(c) To follow a path in a CAM, one starts at a node and traces a line (red) to the right until an edge is hit, then one traces a line down until a node is hit again.

be followed by tracing the underlying thick lines through nodes, which is usually easy in node-link diagrams as well, but not in standard adjacency matrices.

3.4. Approach

Though CAMs have a tidy appearance and standard adjacency matrices are easy to derive from CAMs, the conversion of a network to a CAM is not trivial and consists of six steps. First, the network is decomposed into weakly connected components, after which every component is treated as a separate network for the remaining steps. Second, nodes with identical neighborhoods are grouped. Third, strongly connected components are detected and grouped to form a DAG. Fourth, the nodes of the DAG are partitioned into layers such that all edges have the same direction with respect to the layers. Fifth, the layers are turned into blocks that form the backbone of the CAM, where nodes are partitioned into five classes that dictate their arrangement in the CAM. In addition, grouped strongly connected components are split into separate nodes again while maintaining the vertex arrangement dictated by the blocks. Finally, the blocks are concatenated to form a cascade from which node positions and the CAM visualization are derived in various styles. We describe each step in detail and illustrate it with a running example.

3.4.1. Network

The network is a directed graph $G = (V, E)$, where V is the set of vertices (nodes) and E is the set of *directed* edges between vertices of G. We assume that G is weakly connected, i.e., every vertex is reachable from any other vertex when edge directions are ignored. If a network is not weakly connected, it is decomposed into weakly connected components with a connectivity search. Each weakly connected component is then taken as G and converted to a CAM (see Fig. 3.4). The individual CAMs are concatenated in the final visualization.

The vertices in our network have a number of attributes, such as the gene's name and annotation of gene function. The edges have a single attribute that describes the type of interaction between a gene and its target, namely *promotion*, *inhibition*, *both*, and *unspecified*. However, the exact nature of the attributes is irrelevant to the construction of a CAM.

3.4.2. Identical neighborhood grouping

Biological networks contain many vertices with an identical neighborhood. Grouping these vertices reduces the complexity of the network, while still leaving open the possibility to properly convey each vertex of a group in the final visualization. To this end, we define a grouped version $G_I = (V_I, E_I)$ of G, where V_I is the set of vertices of G_I that represent non-overlapping subsets of V with identical neighborhoods, and E_I is the set of directed edges of G_I.

The edge attribute (describing the type of regulation) is taken into account: only edges with the same value for the attribute are combined into one. An example is shown in Fig. 3.5, in which the vertices e and f both receive input from vertices b and

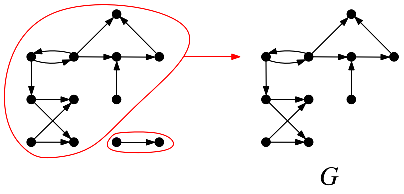


Figure 3.4: *Example network (left) with its weakly connected components marked in red. In this case, the largest component is taken as G (right).*

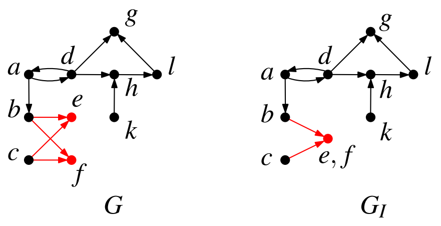


Figure 3.5: *Example of G (left) and the G_I (right) that is derived by grouping vertices with identical neighborhoods. The vertices and edges in red are joined into a larger group.*

c, thus their neighborhoods are considered identical. Such vertices are stacked on top of each other in the final visualization, cf. Fig. 3.3(b).

3.4.3. Strongly connected component grouping

We need to remove all cycles of G_I to turn it into a Directed Acyclic Graph. Along the same lines of the Sugiyama algorithm [38], we could temporarily invert those edges that break a cycle, to be reverted again in the final visualization with a special edge depiction. However, a better alternative is available to us because we know that G and therefore G_I has few and small cycles. The small cycles imply the existence of small *strongly connected components (SCC)*, i.e., sub-graphs where any vertex is reachable from any other vertex.

The SCCs of G_I are a partition of V_I. By grouping every SCC into a single vertex, as shown in Fig. 3.6, we create a DAG $G_S = (V_S, E_S)$. The set of vertices V_S represents SCCs of G_I, and the set E_S is the set of directed edges between vertices in G_S. The edges E_S are not necessarily the same as the edges E_I. They represent an aggregation, which means that edges of E_I with different attribute values can be represented by a single edge of E_S. This is not important for the construction of the CAM, because it relies only on the structure of G_S.

3.4.4. Layers

Graph G_S is a DAG and can therefore be regarded as a layered structure (see Fig. 3.7). We determine its layers L_i, $i = 0, 1, \dots, m$, which form a partition of V_S, in the same way as Sugiyama's algorithm [38]:

The edges that run between layers all have the same direction. Also, no edge runs between two vertices of the same layer. The vertex configuration is as compact as possible: if a vertex $v \in L_i$ has outbound edges, then there exists $v' \in L_{i+1}$ and an edge from v to v', or there exists $v' \in L_{i-1}$ and an edge from v' to v. In other words, vertices

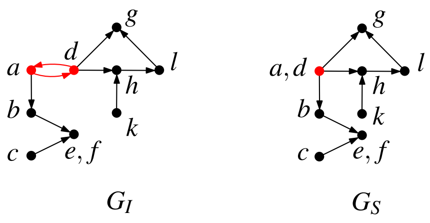


Figure 3.6: *Example of G_I (left) and the G_S (right) that is derived by grouping SCCs. The red vertices and edges are part of an SCC that is larger than a single vertex.*

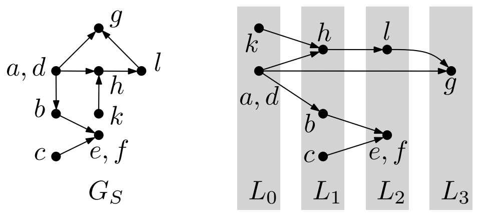


Figure 3.7: *Example of G_S (left) and its layering (right).*

immediately precede their successors in the DAG if there are any successors, or immediately succeed their predecessors otherwise. This is important for the derivation of blocks from layers in a later stage.

3.4.5. Blocks

The layered node-link diagram of Fig. 3.7 already conveys the structure of the network quite well. However, for large networks, the many edge intersections between layers make it hard to interpret. We therefore transform the layers into blocks, which we arrange to obtain a compact visualization.

Layers map directly to blocks, i.e., a block B_i is derived from its corresponding layer L_i. Block B_i consists of sequences of vertices, H_i and V_i, that specify the horizontal and vertical ordering of L_i's vertices in the CAM, respectively. An appropriate ordering of the vertices in H_i and V_i is crucial to obtain the desired compactness. We achieve this by partitioning the vertices of L_i into five classes:

Leaf Vertex in L_i without successors;

Short root Vertex in L_i that has a successor but no predecessors and all successors are leaves in L_{i+1};

Long root Vertex in L_i that has a successor but no predecessors and is not a short root;

Short hub Vertex in L_i that has a predecessor and successor, and all successors are leaves in L_{i+1};

Long hub Vertex in L_i that has a predecessor and successor, but is not a short hub.

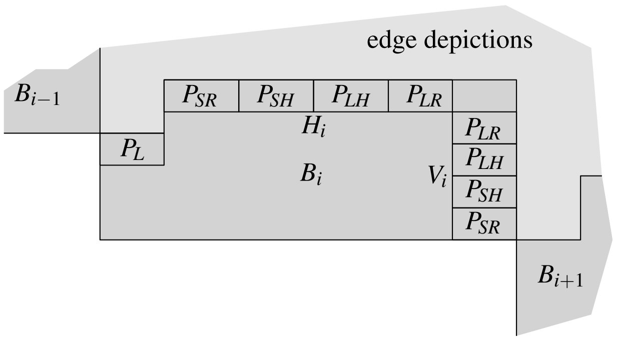


Figure 3.8: *The composition of block B_i and its placement with respect to its neighboring blocks.*

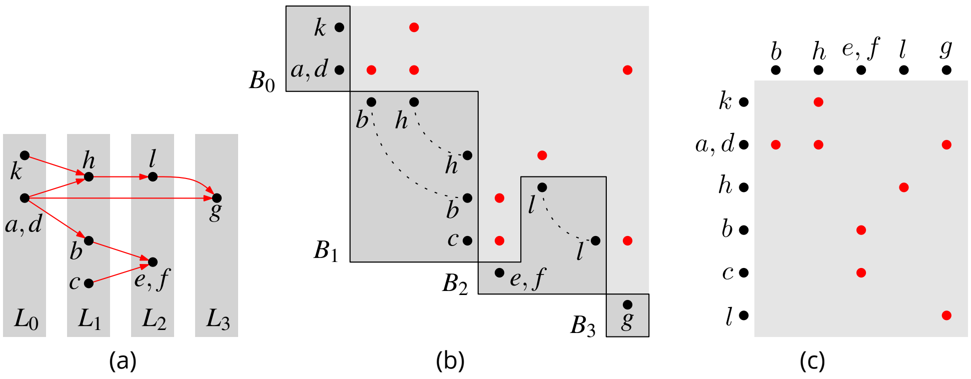


Figure 3.9: *(a) Layers of G_S. (b) Blocks of G_S that form a CAM. (c) Standard adjacency matrix arranged according to the blocks of G_S. All edge depictions are in red. Here e,f and g are leaves, c is a short root, k and a,d are long roots, b and l are short hubs, and h is a long hub.*

The partition that we get for L_i then consists of leaves P_L, short roots P_{SR}, long roots P_{LR}, short hubs P_{SH}, and long hubs P_{LH}. Each of these sets can be ordered individually, based on attributes or some other characteristic. This ordering affects the readability of the CAM, and we come back to this in Section 3.4.6. Once we have the partition, we can construct H_i and V_i by concatenation:

$$H_i = [P_L, P_{SR}, P_{SH}, P_{LH}, P_{LR}]$$
$$V_i = [P_{SR}, P_{SH}, P_{LH}, P_{LR}]$$

The construction of H_i and V_i allows us to make a spatial configuration of B_i as illustrated in Fig. 3.8. The blocks $B_1, B_2, ..., B_m$ form a cascade with enough space above it for edge depictions (see Fig. 3.9(b)). Most vertices of L_i occur twice in this configuration. The vertices of H_i have depictions above them to represent inbound edges from predecessors (in the light gray zone). Similarly, vertices in V_i have depictions to their right to represent outbound edges to successors. This explains why P_L is missing from V_i: it consists of leaves that have no outbound edges.

The presence of P_{SR} in H_i is optional. Removing it will result in a more compact but less consistent CAM, which we discuss in Section 3.4.6. Likewise, the configuration of B_{i-1} and B_i is made more compact by shifting down P_L. This is possible, because vertices of P_{SR} and P_{SH} in B_{i-1} have only outbound edges to P_L of B_i. Therefore, no space is required for edge depictions beyond P_L.

From the arrangement that we now have, we can actually construct a standard adjacency matrix, see Fig. 3.9(c). This can be done by shifting the vertices that belong to some H_i upwards to a common horizontal axis, and shifting those vertices that belong to some V_i to a vertical axis at the left (while respecting their block-induced arrangement).

Finally, the grouped strongly connected components V_S are flattened within H_i and V_i. The sets of identical neighborhood vertices are maintained in the layout, and drawn as stacks of vertices. In addition, edges in E_I that connect vertices of a strongly connected component are represented by arcs, which forms the cycle shown in Fig. 3.3.

For the small example network, clearly not much compression can be achieved. However, Fig. 3.1 and 3.11 show that our approach is effective for compressing large GRNs.

3.4.6. Visualization

Our approach creates a layout that consists of inbound and/or outbound positions for every vertex if it has an in and/or outbound neighborhood, respectively. This provides us with enough information to create a CAM visualization. As in Fig. 3.9, edge depictions are placed in a grid, outbound from the vertex to its direct left and inbound to the vertex directly below it. We give a special treatment to hubs because they have inbound and outbound edges by definition. In standard adjacency matrices, they therefore appear both at the left and at the top of the matrix, spaced wide apart, making it difficult to trace a path. In contrast, we place each hub at the middle of an arc that extends from the hub's inbound neighborhood to its outbound neighborhood. This special treatment of hubs is enabled by their carefully chosen arrangement (cf. Fig. 3.8).

Styles Like for a standard adjacency matrix, the grid drawn in the background is a visual aid that more strongly associates a vertex depiction with edge depictions of its in- and/or outbound neighborhoods (and vice-versa). Even with this aid, adjacency matrices are hard to interpret when they are large. We have therefore experimented with various visual styles to improve interpretation. A sample of style combinations is shown in Fig. 3.10.

The first style is a plain grid, where grid cells are separated by solid lines (see Fig. 3.10(a)). These lines help to inspect the neighborhood of a vertex, but they have to be given a dark color to be visible due to their small width. This creates many strong brightness transitions in the visualization, impeding its aesthetics.

The second style circumvents the need for explicit lines by introducing gray edge depictions for those vertex pairs that have no edges between them (see Fig. 3.10(b)). This generalization creates a grid pattern that guides the observer as well. However, the presence of these non-edge depictions draws away attention from the actual edges, making it harder to get an impression of the network's connectivity. Moreover, a connection between the neighborhoods of hubs still has to be made explicit, which is done with a thick arc that does not integrate well with the visual style of the rest of the CAM.

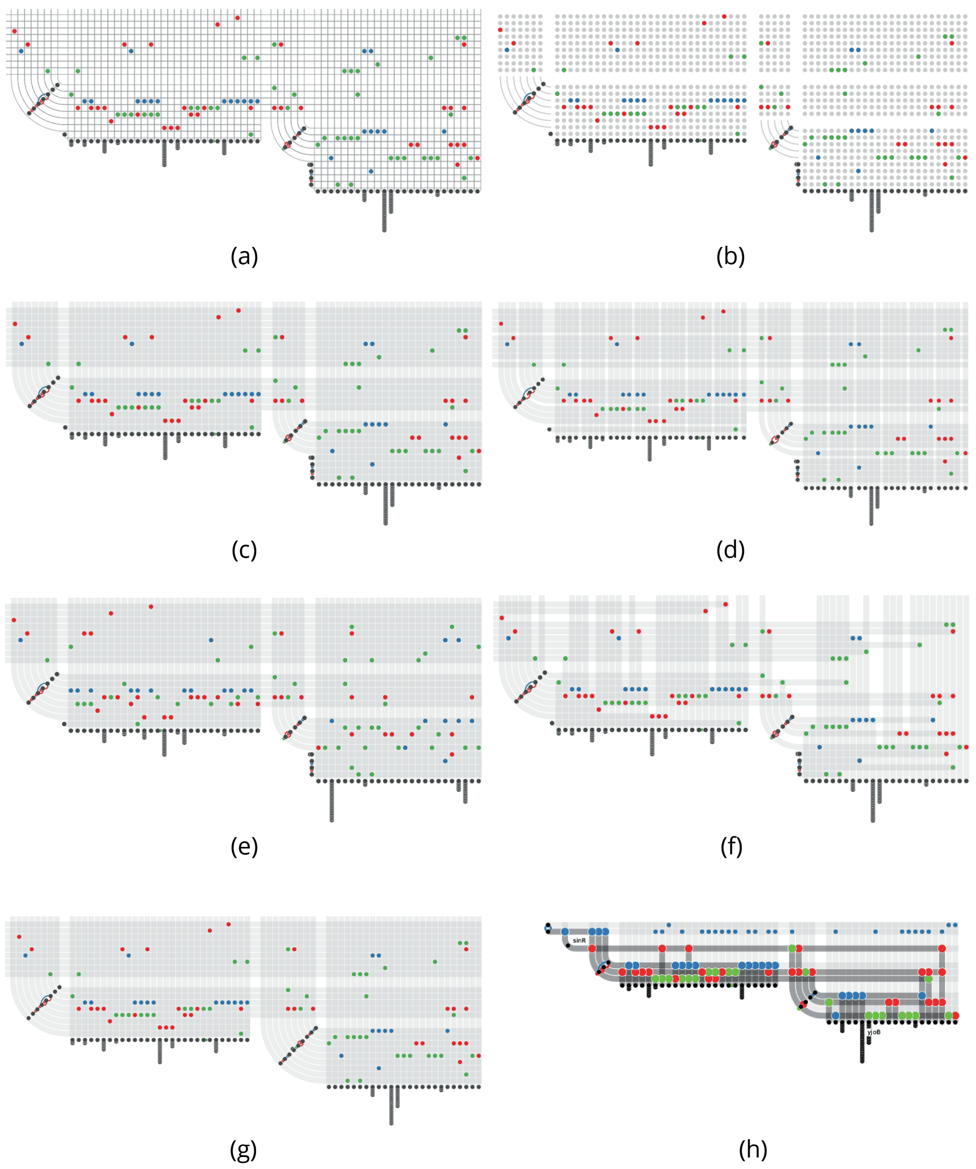


Figure 3.10: *A sample of CAM configurations, applied to a section of Fig. 3.1: (a) Thin separating lines. (b) Non-regulations as gray dots. (c) Thick underlying lines. (d) Periodic spacing. (e) Leaves have no rearrangement clustering on neighborhood similarity. (f) Thick underlying lines are shortened. (g) Roots are treated as hubs. (h) Filtered and highlighted on two selected genes,* sinR *at the left and* yjoB *at the right.*

In our opinion, the third style is the most effective and therefore default style of the prototype. Instead of drawing lines between the edge depictions, thick and translucent lines are drawn behind them (see Fig. 3.10(c)). Vertical and horizontal lines appear darker where they intersect due to their translucency, as though they are plies that are stacked on top of each other. This makes every line of a vertex stand out and better to trace, e.g., it is easy to follow a line down from an edge, along an arc, to the depiction of the hub that the edge is directed to, and even further along the arc, into the hub's outbound neighborhood. Thus, paths in the network are visually apparent and can be followed. Note that this is not possible in a standard adjacency matrix. The other styles also enable path following, but the neighborhood of a vertex does not appear as a clear contiguous area.

Even with the aid of a grid, it is still hard to find all edges of a vertex's neighborhood because it is easy to accidentally skip a line over a long distance. This is similar to drifting off direction when there are no lines guiding you. One way to suppress this effect is to add more structure to the grid, providing additional landmarks to guide an observer. A periodic change of grid cell color is commonly used to get this effect in adjacency matrices and tables. However, we want to restrict the use of colors, because we already use colors to encode edge types. Instead, extra space can be added between vertices to create a visual grouping of cells. This makes skipping a line harder, but also causes the visualization to be less tidy (see Fig. 3.10(d)). Likewise, space between blocks and different classes of vertex can be added if desired.

We have also experimented with minimizing line length while making sure the lines still contain all edges (see Fig. 3.10(f)). This makes it easier to determine the extent of a neighborhood. It also makes the visualization less structured, and implies the presence of edges (possibly non-existing) at line intersections, because the immediate neighborhood is darker. Similarly, the inbound neighborhood line of a root is hidden by default because it wastes space (it has no edges to guide the observer to). Including these lines may be desirable, however, to get a more consistent visualization (see Fig. 3.10(g)).

Color All visual components of the CAM are gray scale except for the edges. The number of grays used is kept at a minimum, because this draws attention to the edges. This also leaves the rest of the color space available to color code edge types such that they are easy to distinguish. Green and red colors encode regulations that promote and inhibit, respectively, because it is a custom in the target domain. Orange encodes regulations that both promote and inhibit, because it is approximately an intermediary for green and red, yet can still be distinguished as a separate color. Blue encodes unspecified regulations, because it has a neutral connotation.

Vertices are colored dark gray, not black, to make them less dominant. They are also given halos matching the background color (white) to improve contrast, and to make them distinguishable when partially stacked. Likewise, edges and text labels are given halos.

Arrangement Various vertex arrangements are implemented in the prototype, i.e., the sets in the partition of a layer L_i (cf. Section 3.4.5) can be ordered in various ways. Simple arrangements like sorting by in- and out-degree are possible, in addition to rearrangement clustering by neighborhood similarity. Similarity clustering is widely used to bring out edge patterns in standard adjacency matrices [62], and by comparison of Fig. 3.10(c) and (e) it is also beneficial to CAMs of GRNs. The use of arrangements is configurable in the prototype, but by default we arrange leaves by inbound neighborhood similarity, and hubs and roots by out-degree. Switching arrangements can be done interactively, where vertices transition smoothly to their new positions.

Vertex arrangement in CAMs is more restricted than in an adjacency matrix due to vertices being part of blocks and their partitions. It is also possible that leaves with similar neighborhoods end up in different layers of the DAG and thus in a different block because one inbound edge is different, causing a large distance between the leaves in the final visualization.

Interaction Hovering over vertices and edges is possible for highlighting and obtaining additional information, i.e., gene name and function description for hovered vertices, and inbound and outbound gene names for hovered edges. In addition, two types of connectivity search are supported: highlighting of direct neighborhood and highlighting of the entire up- and down-directed section of the network, i.e., all genes that (in-)directly influence or are influenced by the hovered gene (see Fig. 3.10(h)). Moreover, multiple vertices can be selected such that any highlighting is kept in place when the vertices are no longer being hovered.

When a vertex or edge is highlighted, its size, color saturation, and brightness is increased, such that it stands out from its surroundings. The background grid lines are darkened as well to provide extra guidance. As shown in Fig. 3.10(h), the enlarged edges also help to distinguish between those edges that are part of the connectivity search, and those that are part of the neighborhood of a connected vertex.

Highlighted vertices and edges attract attention, but the remainder of the network may inhibit their inspection. The prototype therefore allows to filter those vertices and edges that are not highlighted. This causes a considerable reduction in size of the CAM, making it easier to interpret (see Fig. 3.10(h)). Moreover, it enables interactive navigation of the network, because nodes can be added or removed from the selection, increasing or decreasing parts of the network that are highlighted and therefore visible. To better facilitate this navigation, the transition between filtered CAMs is animated, and the underlying block arrangement is maintained to preserve the observer's visual orientation. This means that the entire network is always the basis for the generated CAM, so the DAG structure that leads to layers and blocks is stable, and vertices do not switch blocks, regardless of filtering.

3.5. Discussion

We now compare CAMs to standard visualization techniques that are often used for GRN analysis. Fig. 3.11 and 3.12 show the CAM and the node-link diagram of a GRN of the bacterium *Escherichia coli*, which consists of approximately 1300 genes and 2800 regulations [123].

Node-link diagrams are more intuitive than CAMs, but in case of Fig. 3.12, there are so many edge intersections that the diagram is hardly readable. Some depictions are even fully occluded, leading to a loss of information. More advanced node-link encodings [124], or layout techniques that are specific to scale-free networks [125], may create more insightful visualizations. However, the presence of many edge intersections cannot be avoided due to the interconnectedness of the GRN. Edge-bundling techniques could alleviate this further [22], but at the risk of additional information loss.

The main disadvantage of CAMs is inefficient use of space. Node-link diagrams can be more efficient in this aspect, because the positioning of nodes is less restricted. Even after compression, it can be seen in Fig. 3.11 that large sections of the CAM remain unused, and that it has a high aspect ratio. However, the neatly-arranged visualization and lack of edge overlaps provided by a CAM outweigh these disadvantages.

GRN analysts have specific needs that have to be taken into account as well, which involve the search of patterns. These patterns come in the form of subnetworks with a specific structure. Some instances of these subnetworks are called *motifs*, which are subnetworks that are statistically over-represented in GRNs, and are known to have a specialized function [34]. We therefore consider the most important subnetwork structures, corresponding motifs, and related generic network tasks [9].

Genes should be regarded as part of a dynamical system to understand why specific subnetwork structures have specialized function. Genes have a so-called *level of expression* that is conceptually related to the extent at which the gene is involved in protein production or the regulation of other genes. It is also possible to talk about *regulation signals*, because regulations are dependent on gene expression levels and are therefore time dependent. The motifs that we discuss can thus be regarded as signal processing components, acting as signal delays, filters, and pulse generators.

Self-edge Single vertex with an edge directed to itself.

This structure is of particular interest in GRNs because a self-inhibiting gene (or the *negative auto-regulation* motif) likely shows a faster response to inbound regulation signals, and is more stable when receiving fluctuating signals [34]. Likewise, a self-promoting gene (or the *positive auto-regulation* motif) shows slower response to inbound signals. Self-edges are easily spotted in both Fig. 3.11 and Fig. 3.12 as curves

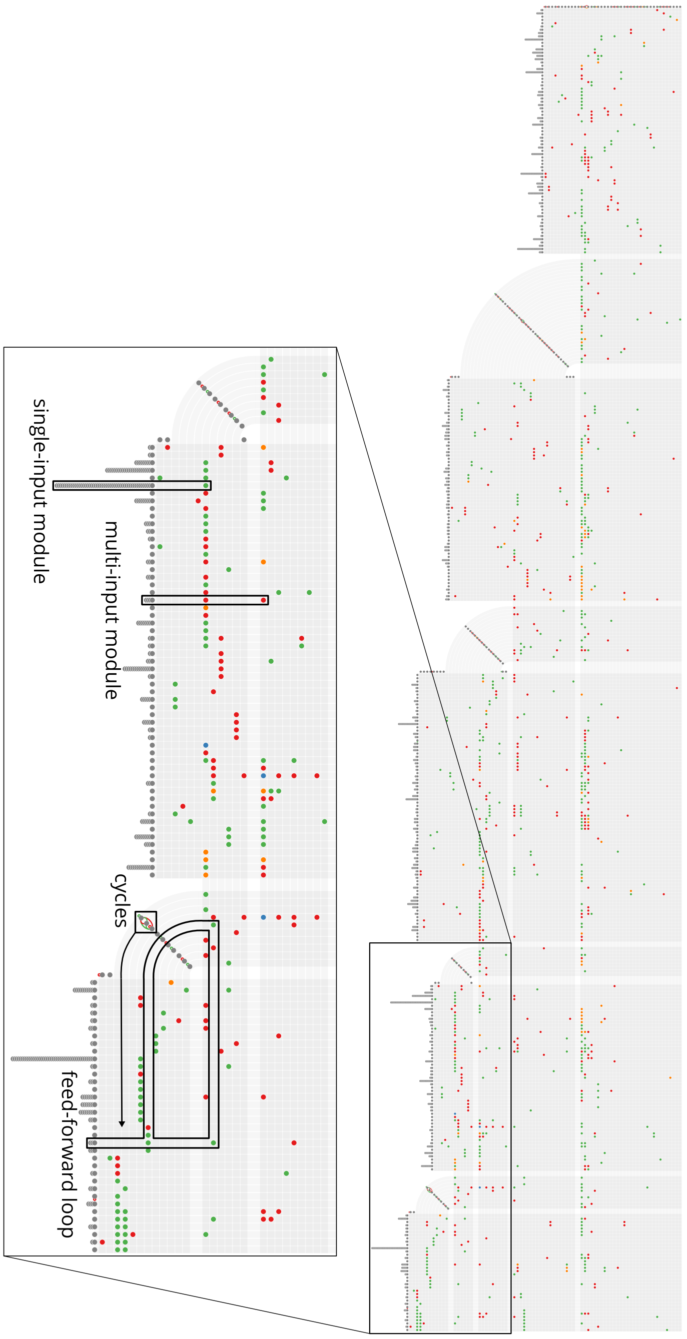


Figure 3.11: *Rotated CAM of a GRN of bacterium Escherichia coli.*

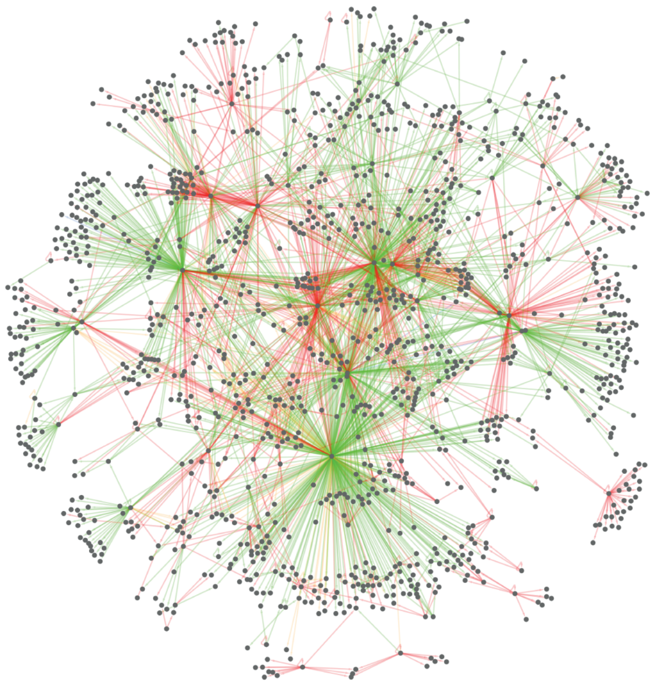


Figure 3.12: *Node-link diagram of a GRN of Escherichia coli, laid out with a spring-based method in Cytoscape as commonly used by GRN analysts.*

that bend back into a node. This also makes self-edges localized such that they are easily spotted as a part of another subnetwork of interest.

Out-fan Vertex with edges directed towards a set of vertices.

Inspecting an out-fan of a vertex requires finding its neighborhood, the *adjacency* task of [9]. This is relevant for gaining insight about *regulators*, which are genes with a large outbound neighborhood, referred to as a *regulon*. These regulators play a dominant role in a bacterium's response to environmental conditions, acting as master switches for parts of a GRN. A regulator is easily spotted in a node-link diagram by the many edges that converge at its position, creating the appearance of an actual fan. Likewise, a regulator is easily spotted in a CAM because it is either a root or a hub, and these vertex classes are easily distinguished. A regulator tends to be in a higher layer and has large strips of edge depictions to its right that is emphasized by neighborhood similarity clustering. In addition, the ordering of hubs and roots by out-degree places regulons close to each other per layer, making them easier to spot. Surveying the entire neighborhood of a regulon, however, is difficult in node-link diagrams because

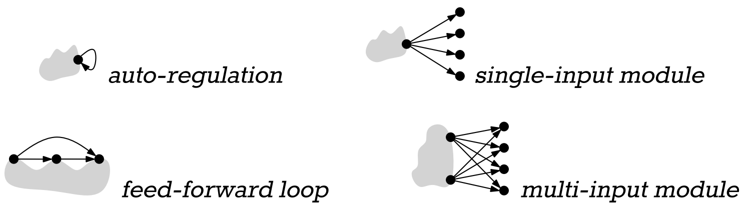


Figure 3.13: *Illustrations of GRN motifs, generalized of regulation types. Gray areas represent the remainder of a network.*

direct neighbors can be positioned anywhere, and individual edges are hard to spot because of the many edge intersections. While neighbors have more regular positions in a CAM, making them easier to find, they can be spread out over longer distances as well.

The *single-input module* motif is a form of out-fan where the genes of a regulon are affected by only one regulator (see Fig. 3.13). Here, a temporal arrangement of gene activations occurs by varying the regulation strength to each gene of the regulon, in essence creating a sequential program that is executed when the regulator becomes active. Single-input modules are spotted in a node-link diagram by their dense arrangement around a regulator, provided that it is not obfuscated by edges that are not part of the module. Stacked leaves in a CAM, with only one inbound edge, correspond directly to a single-input module. For example, it can be seen that *Escherichia coli* has many such modules of varying size spread out over all layers, while *Bacillus subtilis* (see Fig. 3.1) has relatively few large modules that are mostly part of the first layers.

In-fan Vertex with edges directed towards it from a set of vertices.

The observations for the out-fan are symmetric to those of the in-fan. However, the in-fan is of relevance to the *multiple-input module* motif, which is an extension of the single-input module where multiple regulons have strong overlap (see Fig. 3.13). This enables the execution of different sequential programs with the same genes. The special arrangement of vertices in CAMs clearly reveal multiple-input modules, where the out-degree ordering of roots and hubs pushes regulators close together and the neighborhood similarity arrangement of leaves show overlap, and differences, between regulons as thick stripes of edge depictions. Regulon overlap can be detected in node-link diagrams as two fans that diverge to the same set of nodes if only two regulators are involved. However, involvement of more than two regulators results in patterns that are hard to discern. Moreover, it is harder to compare two or more regulons in the node-link diagram of Fig. 3.12 than in the CAM of Fig. 3.11.

Path Multiple vertices connected such that they form a directed chain.

A path in a GRN is effectively an indirect regulation of the gene at the end of the path by the gene at the beginning of the path. Likewise, a regulator may have greater effect on the entire GRN than initially estimated from its regulon, because it affects even more genes via paths through its regulon. This stipulates the desire for focus (e.g., fans) and context (e.g., paths). Tracing a path is called *follow path* in [9].

There may exist multiple paths between two genes. Such regulatory paths tend to pass signals at different speeds, especially when the lengths of the paths are unequal. The relevance of this follows from *feed-forward loop* motifs. Feed-forward loops consist of three genes with one direct and one indirect regulation (see Fig. 3.13). Its behaviors are the filtering of pulses (brief inbound regulation signals), the conversion of long inbound regulation signals to outbound pulses, and the shortening or elongation of response to inbound signals similar to auto-regulation.

Following paths is already difficult in the node-link diagrams of Fig. 3.2 and 3.12, and spotting two paths between two nodes even more so. This is not easy in CAMs either, but feed-forward loops can still be discerned because the nodes of the paths follow each other in the layering and therefore create consistent patterns. These patterns are made explicit by highlighting and filtering, as seen in Fig. 3.10(h).

Cycle Multiple vertices connected such that they form a cycle.

Cycles are also known as *feed-back loops* but they are uncommon in GRNs, and therefore not considered to be motifs. However, genes that are part of cycles have strongly associated behavior, because these cycles tend to be small. For example, two genes that inhibit each other form a cycle that functions as a form of indirect auto-regulation on both genes. In that sense, genes that are part of a cycle behave in unison.

The presence of cycles is clearly lost in large node-link diagrams, where the nodes of a cycle can have large distance between them, and the links of the cycles are obfuscated by other links. Cycles are localized in CAMs and have the appearance of actual cycles, making them easier to spot. Moreover, the complementary nature of genes that are part of a cycle also becomes apparent, because their neighborhood edges are placed close to each other in the matrix. This enables easy comparison of their neighborhoods (see the right-most cycle in Fig. 3.11).

Clearly, there are other visualization techniques beyond node-link diagrams that can be considered, foremost of which are adjacency matrices and more advanced visual encodings. Adjacency matrices, however, are almost identical to CAMs but take up much more space and do not facilitate following a path. The more advanced visual encodings focus on networks that have structural characteristics different from those of GRNs. NodeTrix [60], for example, is effective for networks that have somewhat isolated clusters, which does not match the strongly interconnected nature of GRNs that follows from their scale-free out-degree distribution. Moreover, Quilts [43] are designed for DAG-like networks but rely on a tidy layer structure where few edges skip layers. As becomes clear from Fig. 3.11, GRNs have many edges that skip layers, mainly outbound from important regulators, that make Quilts impractical. We have also tried other node-link layout algorithms, but these gave the same, or worse, results as the one shown in Fig. 3.12.

Finally, the question remains whether GRN analysts will value the use of CAMs over customary node-link diagrams. We have shown several CAMs to biologists in an informal setting. While interpreting the matrix encoding posed an initial challenge, the biologists were able to read and appreciate them after a short explanation. However, it is unlikely that CAMs will see day-to-day use by GRN analysts, who are typically interested in small, isolated GRN modules that can be visualized as node-link diagrams with manageable clutter. A more likely use will come from bioinformaticians, who run algorithms on large GRNs to determine gene importance for example. Superimposing the results of their algorithms on a CAM could assist them in associating the behavior of their algorithms with GRN motifs.

3.6. Conclusion

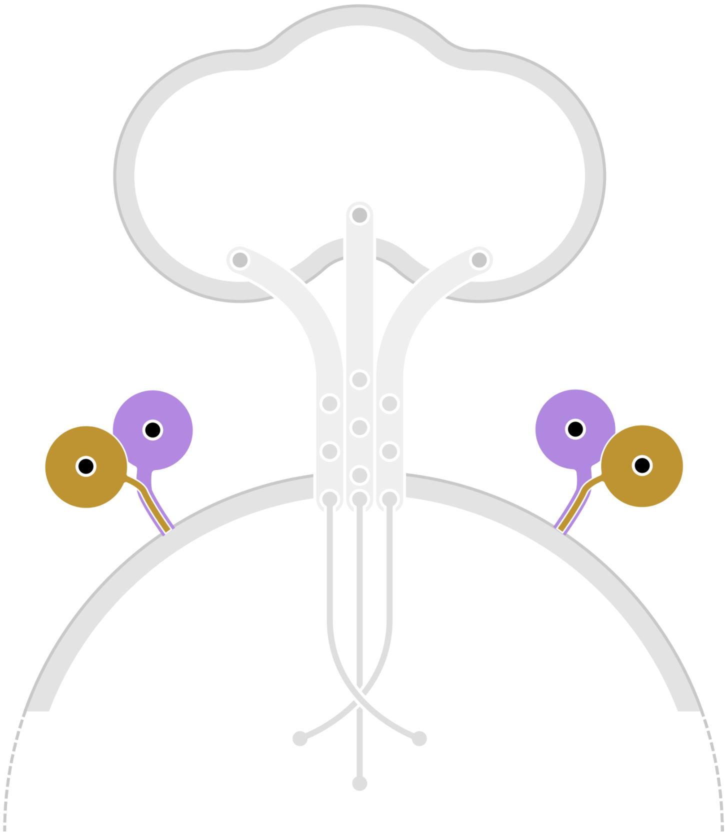
We have presented a new approach for the visualization of GRNs and demonstrated its strengths and weaknesses with respect to finding subnetwork structures that are of importance to GRN analysts. Moreover, CAMs have clear benefits over standard adjacency matrices, such as the ability to follow paths, which in some aspects is even easier than in node-link diagrams. The feasibility of applying the CAM technique in practice is shown with a prototype that supports various interactive techniques often used for standard adjacency matrices. The combination of adjacency matrix specific techniques, such as rearrangement clustering, and node-link diagram properties, such as no node duplication, make for a good alternative to current GRN visualizations.

Future work includes the integration of CAMs into the visual analysis process of GRN analysts. This involves the mapping of additional data to vertices, such as gene expression time series, where we can exploit the linear arrangement of vertices in a CAM to improve attribute comparison tasks. We also want to investigate further matrix compression, for example, by arranging leaves by neighborhood size and pushing sections of them upwards where possible. In addition, the layers of a CAM could be made to branch out, in accordance with a possible branch-like structure of the network itself.

For a CAM to achieve an extensive compression, the visualized network has to fulfill specific requirements. So far we have not encountered other types of network that fulfill these requirements. However, we plan to investigate methods to make CAMs applicable to a broader class of networks. Case in point is an alternate method to deal with cycles, i.e., inverting edges instead of grouping strongly connected components. This means that the few and small cycles requirement can be dropped, but likely at the cost of less intuitive visualizations.

4

Kelp Diagrams



K. Dinkla, M.J. van Kreveld,
B. Speckmann, and M.A. Westenberg

Parts of this chapter have been published in Computer Graphics Forum [212].

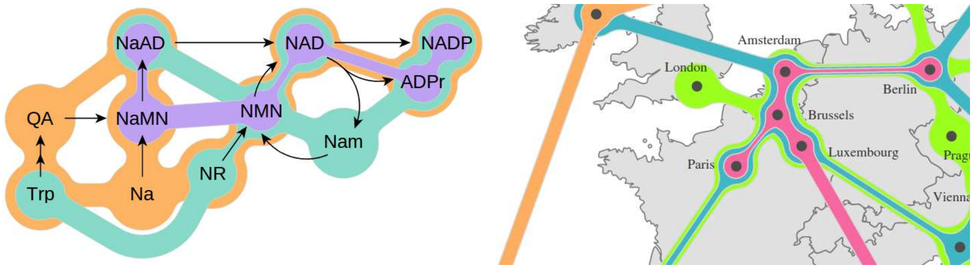


Figure 4.1: *Kelp Diagrams applied to a metabolic pathway (left) and a map (right).*

We present Kelp Diagrams, a novel method to depict set relations over points, i.e., elements with predefined positions. Our method creates schematic drawings and has been designed to take aesthetic quality, efficiency, and effectiveness into account. This is achieved by a routing algorithm, which links elements that are part of the same set by constructing minimum cost paths over a tangent visibility graph. There are two styles of Kelp Diagrams to depict overlapping sets, a nested and a striped style, each with its own strengths and weaknesses. We compare Kelp Diagrams with two existing methods and show that our approach provides a more consistent and clear depiction of both element locations and their set relations.

4.1. Introduction

Visualization of one or multiple sets, possibly sharing several elements, is a recurrent theme both inside and outside the visualization community. Sometimes, depiction of the sets is the main concern, excluding any other information of contained elements besides some means of identification, i.e., a label. Here, simple visualizations suffice for a small number of sets, such as a table with a mapping of elements to rows and sets to columns, or a more space-efficient Euler diagram. Not every situation warrants such an approach. Sometimes, other data aspects (partially) dictate the positions of the elements' visual representations. For example, geographic places are often best positioned at their real-world coordinates because this is consistent with the existing knowledge of the observer and improves visual orientation. Furthermore, it allows spatial patterns to emerge.

The two state of the art approaches, Bubble Sets [126] and LineSets [127], use colored shapes to visually connect elements that belong to the same set. Bubble Sets derives an element density function for each set. Isolines are extracted from each function to form shapes around the elements. These shapes are then connected further with links, routed along the elements. LineSets draws a thick colored curve through the elements of each set, making sure that the traversed path over the elements is relatively short. Both approaches generate visualizations that often appear complex and some-

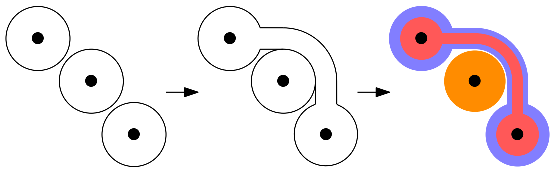


Figure 4.2: *The three phases of generating Kelp Diagrams: element allocation, link allocation, and visualization.*

times even convey invalid set memberships. In contrast our method strictly controls where shapes of sets are placed. It consists of three phases (see Fig. 4.2): the allocation of space around each element such that there is enough room to depict its containing sets; the allocation of space for connecting shapes between demarcation zones with a routing algorithm; and the generation of actual visualizations, by using the allocated space, in two distinct ways.

In summary, our contribution is:

- Two styles of diagram that emphasize different aspects of set memberships and overlap for elements with a predefined position;
- a routing algorithm for linking elements in a set to support the generation of such diagrams, where aesthetic quality, efficiency, and effectiveness are taken into account.

4.2. Related Work

Conveying information about multiple sets is a longstanding problem and exists in various forms. When the location of the elements are not specified, then the input is simply a *set system*. Each set can also be interpreted as a *hyperedge* of a *hypergraph* defined on the elements (the vertices of the hypergraph). There are several papers from the graph drawing community [16, 128] that discuss how to draw hypergraphs. Most recent efforts have focused on so called *planar supports* for hypergraphs and the associated *subdivision drawings* [129] (see also [130-132] for further theoretical results on specific types of planar supports). Unfortunately most hypergraphs do not have planar supports and hence subdivision drawings are of limited practical use.

When only sets (or hyperedges) are of concern, the depiction possibilities are numerous. Euler diagrams are well-known and use contours to denote areas that represent sets, which is sometimes referred to as the *subset standard*. Such diagrams can be generated automatically; by abstracting from individual elements [133], or by including representations of the elements and related information [134-136]. Here, the positions of elements either do not matter as no elements are displayed, or are assigned to optimize the visualization of the sets. The number of elements and sets influences the effectiveness of a visualization. For many elements and groups, augmenting a contour-based visualization is a possibility [137], or a highly-interactive analysis environment is a necessity [138].

When dealing with predefined positions of elements, displaying the sets becomes more difficult. If contours are used, like Euler diagrams, a fair division of display space is an issue [126, 139, 140]. Another option is to connect highlighted elements with (colored) links [90-92]. This is referred to as *visual linking*, which can take on sophisticated forms [93]. However, visual linking focuses on relating elements, not the comparison of sets in a spatial setting. For both approaches the way in which contours or links are placed affects the depiction of the sets but also of the predefined visualization.

4.3. Problem Analysis

An input problem instance consists of three aspects: positioned elements, multiple subsets of these elements, and the predefined visualization that embeds the elements. We want to depict these sets in combination with the predefined visualization. To determine what makes a good set depiction we enumerate tasks that an observer may wish to perform by interpreting the visualization and hence the data. These tasks impose constraints that any visualization has to fulfill whenever possible, but also provide (conflicting) optimization criteria to improve task performance of the observer.

Supported tasks The composition of multiple elements and sets, in a spatial setting, brings forth many questions of a comparative nature: To what extent do sets overlap or differ? How close to each other are the elements of a set? Is the containment of an element in a set correlated to its position? We have compiled a list of primitive tasks that have to be supported by a visualization such that an observer is able to answer these questions, or which the observer can perform ad-hoc to gain insight about the data:

T1a determine the position of an element;

T1b find an element by position (relative to landmarks);

T1c estimate the density of elements in an area;

T2a determine which sets contain a specific element;

T2b find the elements that belong to a specific set;

T2c estimate the spatial distribution of a specific set;

T3 compose a (mental) set from existing sets with operations union, intersect and complement, and apply T1 - T2.

The tasks can be composed to answer more complex questions. When dealing with cities on a map, with a set of *large* cities and a set of *industrial* cities, the following questions may be asked: Which cities are large but not industrial? (T2b and T3); Are industrial cities clustered together? (T2c); Is *New York* considered large and/or industrial, and which neighboring cities are similar? (T1a, T2a, T3, and T2b). As shown, answering common questions involves the combination and execution of these primitive tasks. Thus, improving the efficiency at which tasks can be performed, improves the ease at which complex questions can be answered.

Constraints The visualization should satisfy the following constraints:

C1 every element is clearly represented in the final visualization at its predefined position (for T1);

C2 every element is clearly marked or contained by a representation of every set that it is a part of (for T2).

These constraints are satisfiable, provided that all elements are visually distinct, i.e., all elements are positioned at a discernible distance from each other. We assume this to be the case because any predefined visualization has to support T1 to be of practical purpose and should therefore have visually distinct elements in the first place.

Aesthetic criteria The aforementioned constraints guarantee that all tasks can be performed but do not provide direction towards an efficient visualization that allows (composite) tasks to be performed by an observer with little effort. Generation of aesthetic shapes has been a subject of research before for Euler diagrams [135], and the generation of effective graph representations, by reduction of intersections for example, is a common theme in graph drawing [16]. This has inspired us to list important properties that make for good shapes that depict sets.
Shapes should have low cognitive load:

A1a small area;

A1b few and shallow bends;

A1c few outline intersections.

Not only do aesthetic shapes appeal to the observer, they in general are accompanied by a low cognitive load, i.e., it takes less effort for the observer to process shapes

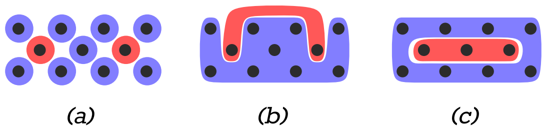


Figure 4.3: *The added benefit of linking: (a) Elements are associated solely by the colored shapes that contain them. (b) Elements are associated both by color and a common shape. (c) Spatial patterns are emphasized.*

and thus derive the information they are meant to convey. For a set depiction, the faster the shapes that convey sets are interpreted, the faster T2 can be performed.

A small area (A1a) implies less surface to inspect and process, thus less cognitive load. Few and shallow bends (A1b) imply smooth outlines that are easy to distinguish from their surroundings (the predefined visualization). It also means that outlines are short and therefore easier to process. Intersections of outlines (A1c) make them harder to tell apart and discern the area they contain and the elements therein.
Shapes should be effective:

A2a large area;

A2b large distance between outlines;

A2c little overlap of shapes that depict different sets;

A2d strong continuation of shapes that depict the same set.

Shapes of large surface area (A2a) attract attention, making the presence of a set explicit (T2). If outlines of shapes are close to each other they are harder to tell apart, often causing the overall shapes to be less pronounced. Likewise, overlapping shapes (A2c) may obfuscate each other and the information they should convey. Both impact T2 negatively.

For T2b (finding the elements that belong to a specific set) to be performed efficiently, the elements have to be associated as a group by the observer. Otherwise, the observer has to scan all elements in a linear fashion to determine their corresponding sets. Incorporating distinguishing features such as color for the shapes is an effective approach. However, creation of a visual continuation of shapes (A2d) causes an even stronger grouping effect (see Fig. 4.3), on which existing approaches rely as well [93, 126, 127]. In certain situations, like the one shown in Fig. 4.3(c), strong continuation also emphasizes spatial patterns of elements and sets. This includes improving the detection of spatial clusters (T2c). As is the case for other criteria, continuation is not a strict constraint, i.e., elements that belong to the same set do not have to be connected.
Shapes should not distort element position and density:

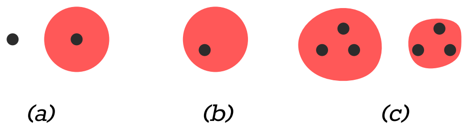


Figure 4.4: *The distorting effect of shapes on element depiction: (a) An element contained in a shape attracts more attention. (b) The expected position of the element lies at the center of the circle, which conflicts with its actual position. (c) Both sets of elements have the same number of elements, but the difference in size of the shapes suggests otherwise.*

A3a little obfuscation of the predefined visualization;

A3b strong correspondence between the presence and size of a set's shape, and the presence and density of elements that belong to this set, in an area.

Not only do the shapes affect the way in which the predefined visualization is perceived through partial occlusion or obfuscation, they also affect the way in which the elements and their locations are perceived (A3a and A3b). For example, when the set containment of an element is depicted with a large colored circle, this circle will form a stronger visual cue to the presence of an element than the element's own depiction (see Fig. 4.4(a)). However, this can also affect the perceived position of the element (see Fig. 4.4(b)) and the density of elements in an area (see Fig. 4.4(c)).

Many of the stated criteria are in conflict with each other: A2a with A2c, A1a with A1b and A1c because of the routing that is required, A1c with A2d, and A3b with all other criteria. Defining an optimal visualization therefore not only requires quantifying the individual criteria, it also requires the criteria to be prioritized and combined into an overall definition of an optimum. Such an approach has been used for the creation of aesthetically pleasing Euler diagrams [133] and label placement on maps [141], where different criteria are weighted and then used as a fitness function. It underlines the varying expectations that different observers may have of visualizations.

Moreover, when criteria like A1c, which require routing, are included in an overall optimization scheme, the combinatorial complexity of the problem greatly increases and forces the use of heuristic algorithms.

4.4. Approach

Our approach consists of three phases: allocation of element space, allocation of additional link space, and the generation of visualizations (see Fig. 4.2). The following pseudo-code provides an overview of the approach, where Sections 4.4.1, 4.4.2, and 4.4.3 elaborate on lines 1 and 2, 3 to 10, and 11, respectively. Here, the elements are denoted as $\mathbb{E}$, the predefined element positions as $p(e)$ for $e \in \mathbb{E}$, and the collection of sets as $\mathbb{S}$, where for every $S \in \mathbb{S}$ we have $S \subseteq \mathbb{E}$.

Algorithm *Kelp*$(\mathbb{E}, \mathbb{S})$

```
1.   Derive Voronoi diagram of $\{p(e)|e \in \mathbb{E}\}$.
2.   For every $e \in \mathbb{E}$ derive $A(e)$ as the intersection of its Voronoi face and a circle
     of radius $r_e$, centered at $p(e)$.
3.   Derive embedded graph $G_A$ from $\{A(e)|e \in \mathbb{E}\}$.
4.   Derive tangent graph $G_T$ from $G_A$.
5.   while best-to-place link $l$ between $p, q \in S \mid S \in \mathbb{S}$ has benefit $b(p, q) > b_t$
6.      do Add edges of $l$ in $G_T$ to subgraph $G_L(S)$.
7.         Derive all-pair shortest paths of $G_L(S)$.
8.         Update $G_T$ with intersections introduced by $l$.
9.         Derive all-pair shortest paths of $G_T$ for all $R \in \mathbb{S}$.
10.        Derive next best-to-place link from shortest paths in $G_L(R)$ and $G_T$ for all
           $R \in \mathbb{S}$.
11.  Derive visualization style from all $G_L(S) \mid S \in \mathbb{S}$.
```

4.4.1. Element space

T2 requires that for each element it is clear to which sets it belongs, hence each element should have ample (and at least equally divided) surrounding space to display its sets. Taking the trade-off between A2a and A2c into account, we want the observer to have a level of control on how much space is used for the set visualization. Given such fixed area per element and considering A1a and A1b, the natural choice of area to allocate around an element is a circle of radius r_e.

It is not always possible to allocate a perfect circle around each element. Sometimes the distance between two elements is smaller than $2r_e$, causing the circles to overlap, or smaller than r_e, causing circles to overlap each other's elements. To resolve this space contention, we first calculate the Voronoi diagram of $\{p(e)|e \in \mathbb{E}\}$ and then, for every element $e \in \mathbb{E}$, intersect the allocated circle of e with the Voronoi face that contains $p(e)$ and use it as the new allocated space $A(e)$ (see Fig. 4.5). Hence, no allocated space intersects and elements get a fair share of space.

The resulting space partition is stored as an embedded graph $G_A = (V_A, E_A)$, with vertices $V_\mathbb{E}$ for the elements, and vertices V_I for the intersection points between Voronoi faces and circles, including the points that are shared by more than two Voronoi faces and lie within an element circle. The edges between these intersection points are either straight (part of a Voronoi face) or circular.

The allocated space of each element is referred to as a *fair* share, not an equal share, because sometimes elements do not receive space that is equal in surface area to their neighboring elements (Fig. 4.6(a)). This unequal allocation could in certain cases affect the depiction of element density (A3b) negatively. Applying a repulsive force between

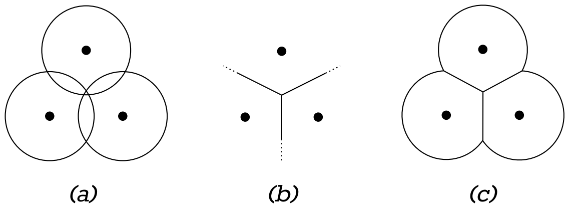


Figure 4.5: *Allocation of space for three elements: (a) Overlapping circles, centered over elements. (b) The Voronoi faces of the elements' positions. (c) Intersection of circles and Voronoi faces.*

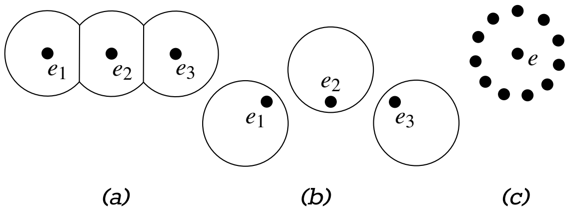


Figure 4.6: *Area division between elements: (a) Our method allocates an unequal share of space. (b) Possible outcome of a force-based relocation of the circles. (c) Instance where allocation of equal space for element e is not possible without a more complex shape.*

the elements' circles, and shifting their positions accordingly, would in many situations result in an equal division (Fig. 4.6(b)), but is not applicable to all situations (Fig. 4.6(c)). Moreover, manipulating the position or shape of the circles will almost always harm the ability of an observer to find elements by position (T1a) and to determine element density in an area (A3b), see also Fig. 4.4. The intersection of an element's Voronoi face and circle is simple and allows for visual reinforcement of the element's position, which is discussed in Section 4.5.

4.4.2. Link space

A2d states that we should visually connect or link those elements that belong to the same set such that elements from the same set are more easily associated with each other. However, any additional link will result in a more complex visualization, negatively affecting most other criteria. Therefore, we have to allocate link space for every $S \in \mathbb{S}$, such that the advantages of the links in the final visualization outweigh their disadvantages, or cost, as much as possible.

We first consider the *cost* $c(l)$ of placing a link l between $p, q \in S$, where $S \in \mathbb{S}$, in the scene. It can be modeled in a simple way, where:

$$c(l) = c_d d(l) + c_\alpha \alpha(l) + c_I I(l)$$

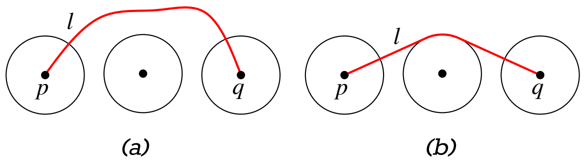


Figure 4.7: *Two possible paths of a link l between $p, q \in \mathbb{E}$ (in red): (a) l with unnecessarily high cost. (b) l with identical topology but minimized cost.*

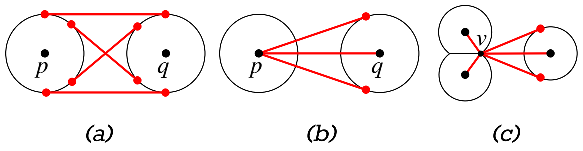


Figure 4.8: *Examples of the additional edges in E_T (in red): (a) Tangent edges between $A(p)$ and $A(q)$, $p, q \in \mathbb{E}$. (b) Edges from p to q and $A(q)$. (c) Edges from $v \in V_I$.*

Here, $d(l)$ is the distance covered by l; $\alpha(l)$ is the aggregate angular change of l in radians; $I(l)$ is the number of intersections between the contours of l and the contours of links already placed in the scene; and c_d, c_α, and c_I are weight parameters. Distance is penalized in accordance with A1a, aggregate angular change with A1b, and intersections with A1c.

No intersections should exist between l and any allocated space $A(e)$ where $e \in \mathbb{E} \setminus \{p, q\}$, as l has no right to use space of $A(e)$. Also, $c(l)$ dictates that l runs directly adjacent to any $A(e)$ that it passes, because tightening l, without altering its topology w.r.t. $A(e)$, will always lower $d(l)$ and $\alpha(l)$ without changing $I(l)$ (see Fig. 4.7).

As can be seen in Fig. 4.7, this tightness means that any desirable link can be constructed from the edges in G_A when certain (tangent) edges are added to it, similar to robot motion planning with tangent visibility graphs [142]. So we extend G_A to form $G_T = (V_T, E_T)$, which includes (tangent) edges between $A(p)$ and $A(q)$, p and $A(q)$, $A(p)$ and q, and p and q. In addition, for every $v \in V_I$, we have edges to every $p \in \mathbb{E}$ and tangent edges to $A(p)$ (see Fig. 4.8). This also adds vertices at the tangent points, which split up the original circular edges.

Now, any link l of set $S \in \mathbb{S}$ can be decomposed into a sequence of touching edges $\{e_1, e_2, ..., e_r\} \subseteq E_T$, and $c(l)$ can then be derived from G_T by summing the costs of the individual edges. Based on these costs, it is possible to compute a minimum cost path between $p, q \in \mathbb{E}$ in G_T and thus get l.

Using only cost as a criterion for placing links is not enough when we want to connect elements beyond a spanning tree. When spanning trees have been established for all sets, additional low-cost links are not always of great benefit to the visualization when they are placed. Consider the situations depicted in Fig. 4.9. For Fig. 4.9(a) the

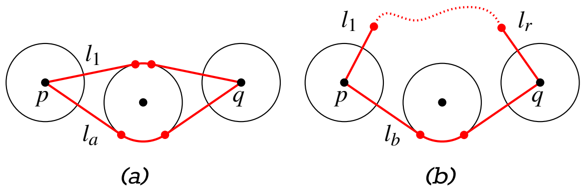


Figure 4.9: *Benefit of placing a link between* $p, q \in S, S \in \mathbb{S}$ *is dependent on already placed links: (a) The benefit of placing* l_a *is low because already placed* l_1 *has low cost. (b) The benefit of placing* l_b *is high because already placed chain of links* $l_1, l_2, ..., l_r$ *has high cost.*

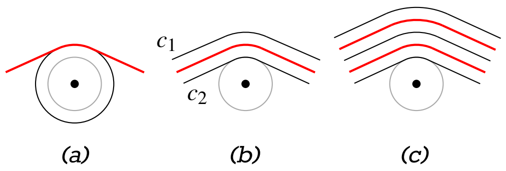


Figure 4.10: *Link radius: (a) Increase of element space allocation by* r_l*. (b) Link with allocated space of radius* r_l *and its contours* c_1 *and* c_2*. (c) Two links routed beside each other.*

benefit of placing link l_a is low because a low cost link l_1 is already in place to provide ample visual linking between the elements. For Fig. 4.9(b) the benefit of placing link l_b is high because the links that already connect p and q sum to a high cost, which means that in the current situation it is hard for the observer to follow links from p to q, while placement of l_b would make this task considerably easier.

Let $G_L(S)$, for every $S \in \mathbb{S}$, be a subgraph of G_T, which contains only edges of links that have so far been added for S. We define the *benefit* of placing a lowest cost link l between $p, q \in S$, as $b(p,q) = \frac{d(p,q)}{c(l)}$, where $d(p,q)$ is the minimum path distance between p and q in $G_L(S)$. Hence, the benefit is higher when the cost of placing l is lower or the smallest distance via already placed links is higher. When p and q are not connected in $G_L(S)$, then $d(p,q) = \infty$. In case the benefit of links has to be compared for disconnected vertices of $G_L(S)$, we compare only by link cost.

Given these definitions, the algorithm places links in a greedy manner: Determine $S \in \mathbb{S}$ and $p, q \in S$ with highest $b(p,q)$. If $b(p,q) > b_t$, where b_t is a benefit threshold parameter, add link l to $G_L(S)$ and repeat the algorithm. When $b(p,q) \leq b_t$, the algorithm terminates.

In our approach so far, routed links have zero width. However, routing links of parameterized radius r_l (width $2r_l$) is achieved by increasing element space allocation from r_e to $r_e + r_l$ (see Fig. 4.10).

When a link is placed, intersection information is updated for every edge $e \in E_T$, such that we know the number of intersections of e's contours (dilation of e by r_l) with contours of already placed links, i.e., dilation of e_l by r_l for all e_l of $G_L(S), S \in \mathbb{S}$. Tracking intersections between just the edges is not sound as it is possible for edges to be placed next to each other within $2r_l$. Links routed over such edges would therefore intersect without receiving the right intersection penalty. To accommodate routing multiple links adjacent to each other and around the same element space (see Fig. 4.10(c)), we also extend G_T to include $A(e)$, and corresponding (tangent) edges, dilated with steps of $2r_l$.

Overlapping contours are not counted as intersections. This allows two links l_1 and l_2 to share part of the same path with few intersections $I(l_1)$ and $I(l_2)$. Low-cost sharing of paths is exactly what we want to properly convey overlapping sets, as explained in Section 4.5.

4.4.3. Visual styles

The space allocated for elements and their sets' visual links can be used in various ways. We devised two very different diagram styles: nested and striped Kelp.

Nested style Nested Kelp surrounds elements of every set with a colored shape (see Fig. 4.11 (top)), where the shapes of every set are stacked on top of each other. It relies on the ability of the observer to mentally distinguish and complete shapes that are partially overlapped by other shapes. The allocated space for elements and links provides a framework to define the diagram such that every shape is sufficiently visible, in correspondence with constraint C2.

We construct the set of shapes $N(S)$ for every $S \in \mathbb{S}$ as follows: For every element $e \in \mathbb{E}$, assume $S_1, S_2, ..., S_r$ contain e, ordered such that $|S_i| \leq |S_j|$ where $i < j$. For every S_i, scale the allocated space $A(e)$ around its centroid by a factor $(\frac{i}{r})^{s_e}$, where s_e is a parameter, and merge it into $N(S_i)$.

For every edge $e \in E_L$, let Q be the set of edges reachable from e in the union of all $G_L(S), S \in \mathbb{S}$, without passing beyond an element node. Assume that $S_1, S_2, ..., S_r$ contain an edge in Q, ordered such that $|S_i| \leq |S_j|$ where $i < j$. Then, for every S_i, dilate e by $r_l(\frac{i}{r})^{s_l}$, where s_l is a parameter, and merge it into $N(S_i)$. Here *dilate* and *erode* are the equivalents of *Minkowski sum* and *Minkowski subtraction* with a circle of a specified radius [143]. Thus, for every element, set membership depictions are bound by the space that was allocated for the element. In addition, depictions are scaled to nest. Links and element circles thus do not become visually dominant when many sets share an element or path.

Links of a set are scaled to nest with links of other sets that partially share a path in G_L. The order in which sets are nested is based on the size of sets, i.e., smaller sets nest in larger sets, which is consistent throughout the diagram. The scaling of both element circles and links, by their level of nesting, depends on parameters s_e

and s_l, respectively, such that the share of space allocated to every set can be adjusted to compensate for the effects of perceptual scaling [144].

The shapes $N(S)$, for $S \in \mathbb{S}$, are drawn on top of each other, ordered by $|S|$, where $N(S)$ is filled with a color and given a gray outline to enhance contrast between shapes of different sets. In addition, the shapes are dilated and eroded to smoothen the corners of the allocated element space and the transition between links and elements. A strong erosion results in a clear separation between shapes that are not nested (A2b), as seen at the top of Fig. 4.11.

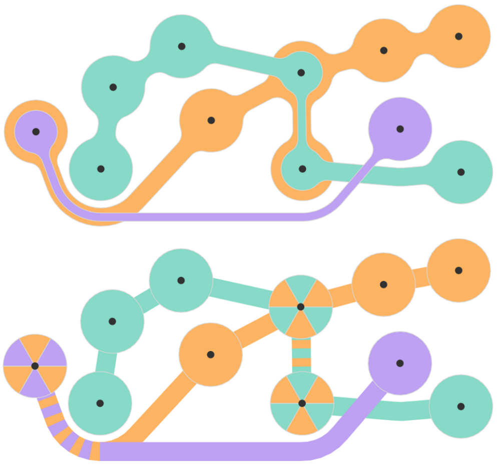


Figure 4.11: *Kelp Diagram of eleven elements and three sets. Top: Nested style. Bottom: Striped style.*

Striped style Striped Kelp uses alternating stripes for areas that contain elements of multiple sets (see bottom of Fig. 4.11). The allocated space $A(e)$ of $e \in \mathbb{E}$ is partitioned into radial slices, where every set that contains e gets the same number of slices in an alternating pattern. Edges of G_L that belong to a set have link radius r_l. When an edge belongs to multiple sets (links partially share a path), it is partitioned into stripes of fixed length where the sets give an alternating pattern.

Stripes are drawn as consistently as possible. If two links share edges but eventually split up, the stripes will continue along the edges at the split. This is achieved by drawing the stripes via a depth-first search in the G_L graphs. Certain cyclic configurations of links do not allow for consistent stripes, but these are rare enough in practice.

4.4.4. Implementation and performance

We implemented our approach in Java, using the Java Topology Suite [145] for geometric operations such as dilation and erosion. One advantage of our purely geometric routing and diagram definitions is that vector graphics can be generated and merged with the predefined (vector) visualization to get images that can be rendered with arbitrary resolution.

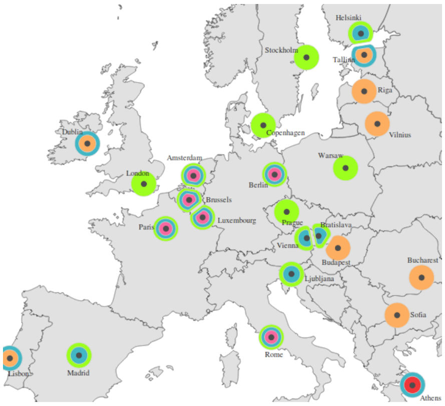
The running time of the algorithm is dominated by the iterative placement of links. The tangent graph G_T contains $O(|\mathbb{E}|^2)$ edges and nodes. To place one link, shortest paths are computed to all nodes in G_T from every element and for every set. The same applies to all G_S graphs. This computation, including selection of the best link, amounts to $O(|\mathbb{S}||\mathbb{E}|^2 \log |\mathbb{E}|)$ when Dijkstra's algorithm is used. After a link is placed, any intersections that it introduces have to be accounted for. Thus, intersections are determined for all edge pairs, which amounts to $O(|\mathbb{E}|^2 \log |\mathbb{E}|)$ time with a sweepline algorithm. It is reasonable to assume that the algorithm's parameters are configured such that for every set $S \in \mathbb{S}$ the final $G_S(S)$ is planar and therefore $O(|\mathbb{E}|)$ links are placed for S. The entire links placement phase therefore takes $O(|\mathbb{S}|^2|\mathbb{E}|^3 \log |\mathbb{E}|)$ time. This also bounds the running time of the entire approach, with the space allocation and visualization phases being relatively cheap.

The bound (though not tight) indicates that the current approach cannot be applied to data sets of thousands of elements. However, most data sets commonly used in this visualization problem do not go beyond a hundred elements and several sets, since larger sets cannot be comprehended by an observer. Our implementation is able to generate Kelp diagrams for such data sets within five minutes on a modern desktop computer (Intel Core 2 Quad CPU at 2.4 GHz).

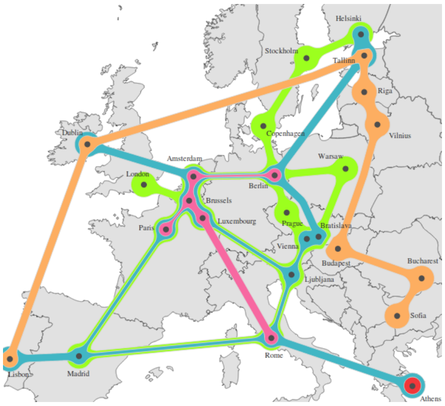
4.5. Results and Discussion

Fig. 4.12 shows Kelp Diagrams with various parameter configurations. The benefit of visually linking elements that belong to the same set follows from Fig. 4.12(a) and (b). Increasing the element radius r_e (see Fig. 4.12(b) and (d)) causes more of the original map to be occluded, making it harder to find capital cities by their location, but sets are more easily distinguished. Especially when multiple links share a path, the sets of the nested style are harder to distinguish, requiring larger r_l. The striped style suffers less from this. A comparison of Fig. 4.12(b) and (e) reveals the effect of increasing the intersection penalty c_I, where the algorithm deems it of greater benefit to route a long link around Helsinki instead of introducing additional intersections. The effect of the link addition threshold b_t is shown in Fig. 4.12(f), where a low b_t results in the construction of spanning trees. Kelp can be applied beyond cartography, as shown for a metabolic network in Fig. 4.1. Here, nested element scaling s_e has been given a very low value to push the contours away from labels of compounds.

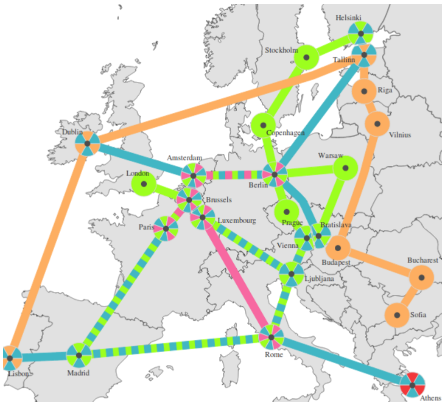
Figure 4.12: *Kelp applied to the capital cities of the European Union, where the eurozone is blue, the EU founding members (European Coal and Steel Community) are pink, and members with good, average, and bad credit rating are green, orange, and red respectively (Standard & Poor's, October 2011). The diagrams have various configurations: (a) Nested style without links. (b) Nested style with links. (c) Striped style. (d) Nested style with large element radius r_e and small link radius r_l. (e) Nested style with large intersection penalty c_I. (f) Nested style with low link addition threshold b_t.*



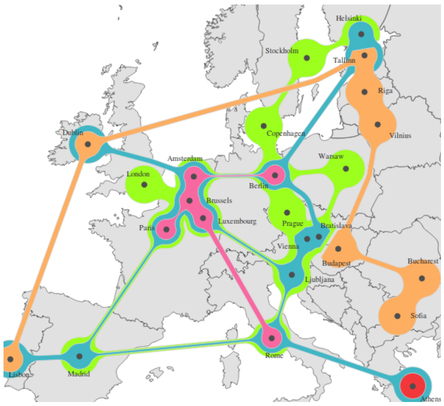
(a) *disconnected*



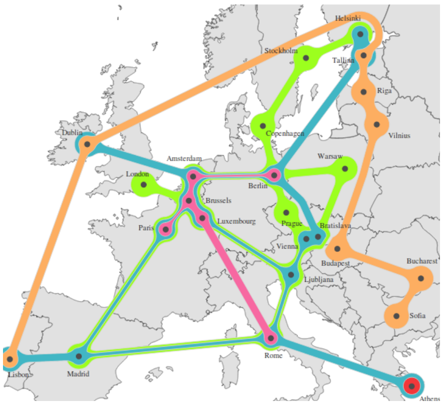
(b) *connected*



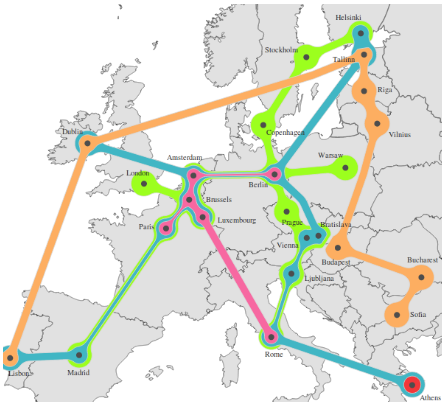
(c) *striped*



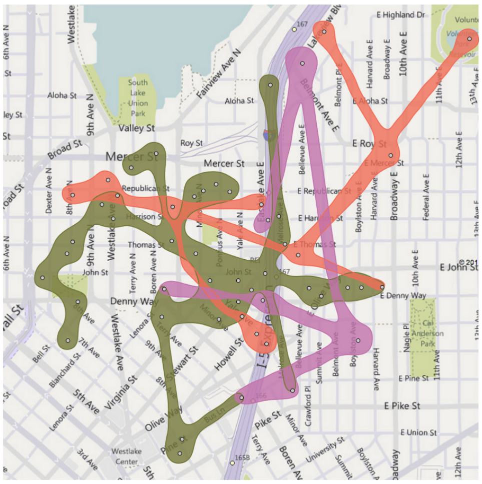
(d) *big elements and thin links*



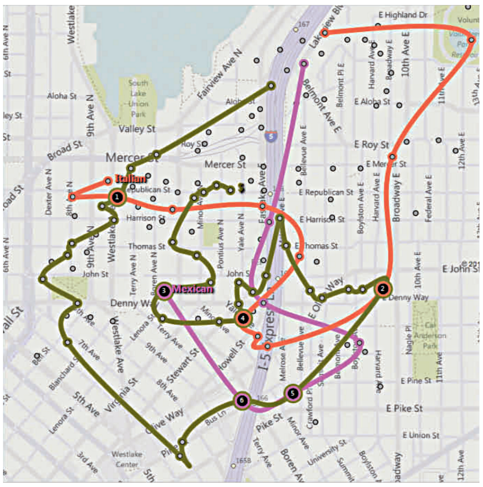
(e) *few intersections*



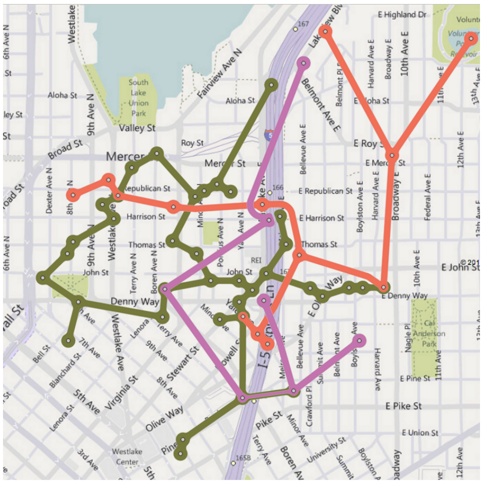
(f) *few links*



(a)



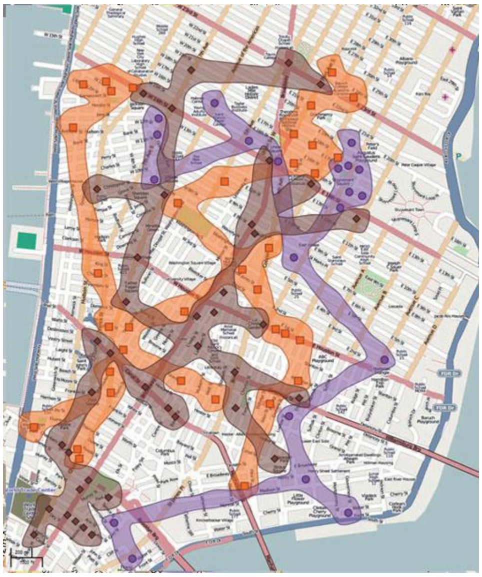
(b)

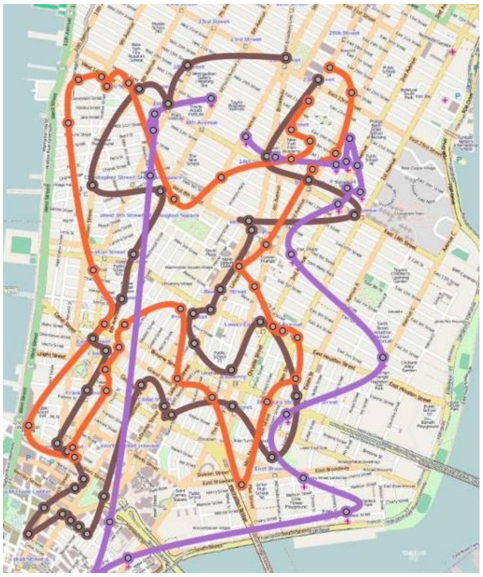


(c)

Figure 4.13: *Restaurants of three categories in Seattle:*
(a) Bubble Sets approach, generated with a public software library [146].
(b) LineSets approach, image reproduced from [127].
(c) Nested Kelp Diagram.

The strengths and weaknesses of the nested and striped styles can be seen in Fig. 4.12(b) and (c). When multiple sets share the same link, e.g., Amsterdam-Berlin, the sets of the nested style quickly become harder to distinguish, whereas the striped style has no such problem provided that the link is long enough to fit ample stripes for each set. The striped style also prevents false assumptions to be made about the nested structure of the depicted sets. The nesting at Ljubljana suggests that the blue set is a subset of the green set, while they actually only overlap. However, the striped style ignores some of the criteria listed in Section 4.3. The stripes of a shared link or


(a)


(b)

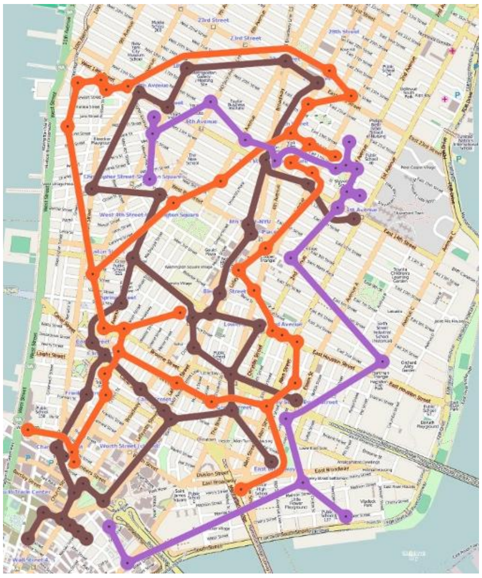

(c)

Figure 4.14: *Disjoint sets of locations in Manhattan, with hotels (orange), subway stations (brown), and medical clinics (purple):*
(a) Bubble Sets, image taken from [126].
(b) LineSets, image courtesy B. Alper and N. Riche.
(c) Nested Kelp Diagram.

node break the continuation of the sets' shapes, which explains why the shapes of the nested style are more easily interpreted as a whole. Stripes also increase the visual complexity of the diagram because there are more and stronger bends (A1b). Therefore, the nested style is easier on the eyes and likely the more versatile style in many situations. It also happens to be the closest visual match to existing techniques.

Fig. 4.13 and 4.14 compare the nested style with the Bubble Sets and LineSets techniques for the same datasets. We can see that the Bubble Sets approach has many bends in its contours. In some locations, shapes and contours have undesirable overlap. Compared to the Kelp Diagram, the Bubble Sets depiction introduces more overlapping areas. This is not surprising since the Bubble Sets algorithm does not try to avoid intersections between links where Kelp does. In addition, Bubble Sets create shapes that use a lot of space and color blending introduces new colors for overlapping shapes that may be interpreted as additional, non-existing sets.

Bubble Sets and LineSets cannot visually connect elements beyond creating a spanning tree, whereas Kelp creates a graph to interconnect elements as much as desired via parameter b_t. Additional links, beyond a spanning tree, may not always be desired. For example, Fig. 4.14(c) has been generated with a relatively high b_t. For this diagram, some links of the brown set introduce some additional visual clutter (A1a, A1b, and A3a) though improve interpretation of spatial distributions (A2d and A3b). However, when b_t is set to a low value, surplus links will only be placed when continuation greatly improves. This is the case for the green set of Fig. 4.13.

LineSets do not route shapes around elements that should not be contained by them, which the other techniques do. Hence, in Fig. 4.14(b) we see an element of the brown set at the middle left that overlaps with a shape of the orange set, while it is not a part of the orange set. Moreover, LineSets connect elements of a set with a single line, creating longer links, many bends, and intersections. For example, there is a line with a strong bend at the bottom of Fig. 4.13(b) that could have been avoided.

Even though the element glyphs are very small in Fig. 4.13(c), we can immediately see the presence and density of elements because the surrounding contours are (partially) circular. For LineSets, the presence of an element can be inferred from the presence of a bend, while for Bubble Sets the large shapes make elements harder to spot (see Fig. 4.13(a)). Kelp's composition of basic visual units, circles and constant-width connections, is an advantage over Bubble Sets. Depictions are consistent, which means less visual clutter and easier interpretation.

Kelp Diagrams have some drawbacks. They sometimes have sharp bends where two links of the same set converge. Also, contours can become complex for clusters of elements that belong to the same set. LineSets suffer from this as well, in contrast to Bubble Sets. In addition, when elements are too close to each other, their Voronoi faces become dominant in space allocation and cause strong corners to occur in nearby contours. The schematic appearance of Kelp on top of a predefined visualization that is schematic as well can be confusing, e.g., the set shapes may be interpreted as roads or metro lines. The more organic appearance of LineSets and Bubble Sets make them stand out from a map.

For biological networks, conventional approaches change the color of compounds to visualize a single region of interest in, for example, a metabolic pathway. However, Fig. 4.1 depicts a Kelp Diagram on top of a metabolic pathway such that multiple properties of compounds are simultaneously conveyed. This flexibility extends to other biological networks such as Gene Regulatory and Protein Interaction Networks, enabling the visualization of multiple overlapping modules. In Chapter 5 we demonstrate how this capability benefits actual network analysis.

4.6. Conclusion

Kelp Diagrams are a new way to depict set relations over already positioned elements. The algorithm balances visual complexity, based on aesthetic criteria, with effective depiction of the data. Comparison of resulting visualizations with those generated by two state of the art approaches for the same data shows that Kelp Diagrams have a consistent, easy to interpret, appearance. In addition, the parameters of the algorithm give it the flexibility that is required for application to different kinds of visualization.

Still, several improvements are possible. The routing algorithm is too slow for interactive use. Exploiting locality with spatial data structures, or replacing the tangent visibility graph with a graph of smaller complexity, are possible options. Also the greedy placement of links can be improved to reduce visual clutter. For nested Kelp, improved derivation of link width and nesting order with respect to aesthetic criteria requires further investigation. For example, links could be given widths according to an additive scheme, where the width of a link is increased when it has to convey multiple sets. In addition, allocated element and link space could be made dependent on the element density in an area.

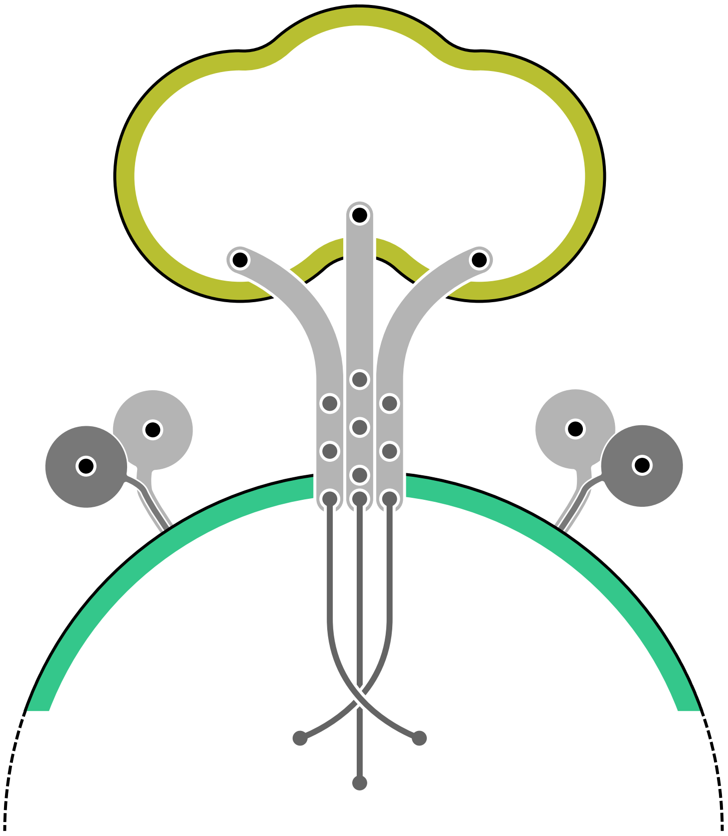
Other extensions are possible, such as supporting elements with dimension (instead of points) and automated derivation of parameter settings based on the data itself. Use of the subset standard, as is the case for Kelp Diagrams, has been indicated to be effective via user studies before [127]. Nonetheless, we plan to investigate the effectiveness of the presented diagrams, and compare them to existing techniques, with further user studies.

Current approaches share many similarities in an attempt to solve the same problem. KelpFusion [147] is a more generic method that is able to produce the distinct visualizations of current approaches, but also able to produce hybrid visualizations of greater quality.

Acknowledgements We would like to thank Nathalie Henry Riche, Basak Alper, and their colleagues for providing us with the data for Fig. 4.13 and 4.14.

5

eXamine



K. Dinkla, M. El-Kebir, C. Bucur, M. Siderius,
M.J. Smit, M.A. Westenberg, G.W. Klau

Parts of this chapter have been published in BMC Bioinformatics [213].

Biological networks have a growing importance for the interpretation of large scale "omics" data. Integrative network analysis makes use of statistical and combinatorial methods to extract smaller subnetwork modules, and performs enrichment analysis to annotate the modules with ontology terms or other available knowledge. This process results in annotated modules, which retain the original network structure and includes enrichment information as a set system. A major bottleneck is a lack of tools that allow exploring both network structure of extracted modules and its annotations. We therefore present a visual analysis approach that targets small modules with many set-based annotations, and which displays the annotations as contours on top of a node-link diagram. We introduce an extension of self-organizing maps to lay out nodes, links, and contours in a unified way. An implementation of this approach is freely available as the Cytoscape app eXamine. This app accurately conveys small and annotated modules consisting of several dozens of proteins and annotations. We demonstrate that eXamine facilitates the interpretation of integrative network analysis results in a guided case study. This study has resulted in a novel biological insight regarding the virally-encoded G-protein coupled receptor US28.

5.1. Background

High-throughput "omics" data provide snapshots of cellular states in a specific condition. Computational approaches can be used to relate these low-level measurements with high-level changes in phenotype. Traditionally, these approaches were *gene-centric* and typically resulted in ranked lists of differentially expressed genes [148-150]. Later, gene-centric approaches were complemented by *pathway-* [151, 152] and *network-based* methods [153, 154] to provide inter-gene context for mechanistic insights. Pathway-based approaches identify overrepresented pathways from databases such as the Kyoto Encyclopedia of Genes and Genomes (KEGG) [155]. Network-based approaches yield small, *de novo* subnetwork modules, which may span several known pathways and reveal their crosstalk [156].

Extracted network modules are analyzed in the context of established gene annotations to hypothesize about the module's role in high-level cell conditions (see Fig. 5.1). Genes are often related to very many terms (too many for human comprehension), most of which are likely irrelevant to the analysis context. Therefore, overrepresentation analysis is performed to rank information items by their significance. These items originate from ontologies such as the Gene Ontology (GO) [157], which identifies cellular functions, processes and components that nodes relate to, or from KEGG [155], which relates nodes to pathways. This results in an *annotated module*, which retains the original network structure and includes enrichment information as a *set system*.

Existing tools focus on visualizing large networks, and have only limited or separate set system support or no support at all. Our proposed visual analysis approach displays sets as contours on top of a node-link layout (see Fig. 5.2). It treats module

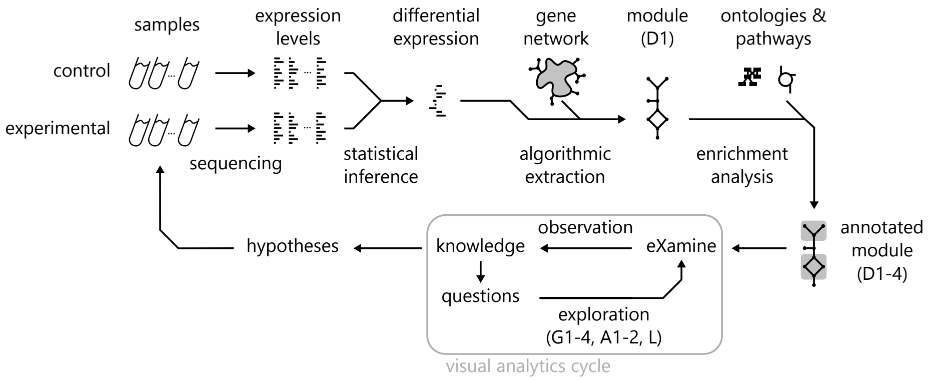


Figure 5.1: *Data and analysis pipeline. First, control and experimental samples are analyzed to estimate expression levels. Subsequently, gene expression differences (between experiment and control) and their significance are determined. These differences are then mapped to an interaction network, from which a module is extracted with overall significantly-differential gene expression. This module is annotated with overrepresented cell mechanisms from ontology and pathway databases. Finally, the enriched module undergoes iterative visual analysis via eXamine.*

edges and annotation sets in a unified way, and contributes the following to the analysis of annotated modules:

- Identification of elementary module analysis tasks and their composition into a visual analysis process;
- Extension of the self-organizing maps (SOM) algorithm to lay out module interactions and annotations in a unified approach;
- Implementation in the form of the Cytoscape app eXamine;
- Demonstration of eXamine via a guided study of an annotated module that is activated by the virally-encoded G protein-coupled receptor US28;
- Discussion on how eXamine facilitates the analysis process.

5.1.1. Data characteristics

The annotated modules, targeted by the presented method, have the following characteristics.

D1 Small and sparse network topology, in which genes and interactions number in the dozens;

D2 Many annotation sets, outnumbering gene interactions;

D3 Annotation sets vary in cardinality, from a single node to the entire module;

D4 Annotation sets overlap often.

Integrative network analysis methods produce small and sparse subnetwork modules (D1), rather than large lists of differentially expressed genes. Embedding the module in a rich context of annotations on overlapping sets of genes is a typical next step to gain insights in the underlying biology (D2, D3, D4).

5.1.2. Analysis tasks

The focus (or perspective) of analysts alternates between genes (and interactions within a module) and annotation sets. Important analysis tasks are supported for each of these data aspects to enable an analyst to hypothesize about the role of an extracted module in light of experimental conditions.

For genes, analysts want to determine:

G1 Level of differential expression: under- or over-expressed, or insignificant;

G2 Interacting neighbors;

G3 Annotations (set memberships);

G4 Annotations shared with other genes.

Single genes can become the focus of attention during the analysis process within the context of the module. The fact that a gene is part of a module does not imply that its under- or overexpression is significant. However, information (G1) about differential expression enables the elucidation of a gene's presence in the module. For example, it could be the case that a gene is not differentially expressed significantly itself, but that it is still part of a module, because it connects two differentially expressed submodules. An indirect involvement of the gene in a module mechanism is therefore likely. Neighboring genes might also become interesting (G2), as are any mechanisms that it is associated with already (G3), and the mechanisms that it shares with other genes in the module (G4).

For annotation sets, analysts want to determine:

A1 Significance of overrepresentation;

A2 Gene memberships.

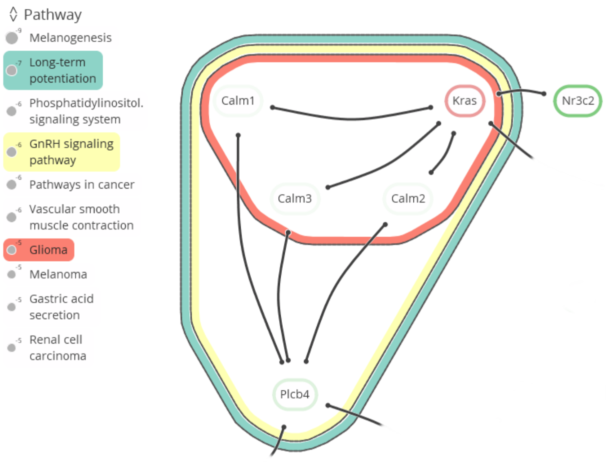

Pathway
-9 Melanogenesis
-7 Long-term potentiation
-6 Phosphatidylinositol. signaling system
-6 GnRH signaling pathway
-6 Pathways in cancer
-6 Vascular smooth muscle contraction
-5 Glioma
-5 Melanoma
-5 Gastric acid secretion
-5 Renal cell carcinoma
Calm1
Kras
Nr3c2
Calm3
Calm2
Plcb4

Figure 5.2: *Visualization of an annotated module. Interacting proteins with a selection of three subsets, corresponding to overrepresented KEGG pathways. The visualization consists of a combination of a node-link diagram and an Euler diagram.*

If a specific gene is interesting, its annotations might be too (G3 and G4). Annotation sets themselves can have such significance (A1) that they become interesting, which then translates to genes contained in them (A2). Both significance in terms of an associated p-value and subjective significance are of importance to divide attention between annotation sets.

For interactions, analysts want to determine:

L Annotation transitions between interacting genes.

A change between annotations (L) may occur when the focus on a gene shifts to a neighboring gene (G2), which is of importance to an analyst to judge the role and relevance of the neighboring gene in the module.

5.1.3. Related work

Network visualization and tools Many advanced techniques for the visualization of network topology have been developed [5, 6, 16], but few have been transferred to readily available tools. On the other hand, there are many tools for interpreting and exploring biological networks [158], including the popular open source platforms Cytoscape [10] and PathVisio [159]. However, these currently provide only limited capability to visualize annotated modules. PathVisio is a pathway analysis approach, in which sets are restricted to subsets of static, pre-defined individual pathways, and set membership is conveyed via node colors. Cytoscape's group attributes layout can be

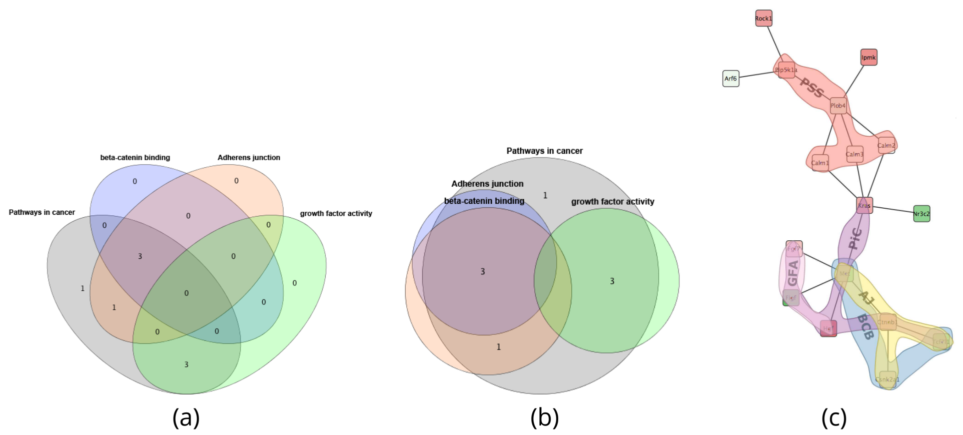


Figure 5.3: *Annotated module visualization using Cytoscape's Venn and Euler diagram app: (a) Venn diagram and (b) Euler diagram. The number of displayed sets is limited to four and no network structure is shown. (c) Module laid out by one of Cytoscape's built-in force-directed layout algorithms and BubbleSets superimposed on the network (same color scheme as in Fig. 5.9(b)). Note that it is not immediately apparent that the nodes in the β-catenin set (blue) form a subset of Adherens junction (yellow), because the BubbleSet approach applies no explicit nesting of subsets.*

used to visualize partitions by showing disjoint parts in separate circles, but it does not support overlapping sets. The Venn and Euler diagram app [160] for Cytoscape does support overlapping sets, but it can handle only four at the same time (see Figs. 5.3(a) and (b)). In this app, network and sets are visualized separately: set membership is conveyed by selecting a set and its corresponding nodes are highlighted in Cytoscape's network view. The RBVI collection of plugins [161] facilitates creation and editing of Cytoscape groups, and provides a group viewer that relies on aggregation of groups into meta-nodes. These meta-nodes can be visualized as standard nodes, as nodes containing embedded networks, or as charts. This approach, however, does not allow for visualization of overlapping sets.

Set system visualization In the information visualization field, *Euler diagrams* are used for the intuitive visualization of set systems [134-136], in which items belonging to the same set are denoted by contours. Variants of these approaches visualize sets over items with predefined positions, e.g., over a given node-link visualization of a network. These methods range from connecting these items by simple lines (LineSets) [127], via colored shapes that are routed along the items (Kelp Diagrams) [212] and contours around the items (BubbleSets, see Fig. 5.3(c)) [126, 162] to hybrid approaches

(KelpFusion) [147]. Visualizing an annotated module, however, requires an integrated layout of both its network and set system topologies, which is not possible with these approaches. Euler diagram methods focus on the layout of set relations at the expense of network topology. Likewise, laying out the network before superimposing set relations will emphasize network topology to the detriment of the set system. Some techniques exist that provide such integrated layouts [59, 90, 163, 164], and which include aesthetic concerns and design of visual metaphors [165]. However, these approaches assume constraints on the network and set system topologies, e.g., strict partitions and no overlapping sets, and they are therefore not applicable to our problem.

5.2. Method and Implementation

Visualizing an annotated module amounts to visualizing a *hypergraph* consisting of binary edges (interactions) between nodes (genes) and n-ary edges (annotation sets). Analysis tasks G2-G4 and A2 establish the equal importance of associating interactions and annotation sets, which reflect on both the layout as well as the visualization of the hypergraph. Therefore, as opposed to combining multiple existing techniques (e.g., a force simulation to position the nodes according to the binary edges [19], a node overlap removal algorithm to keep nodes identifiable [166], and subsequent construction of a density field to derive contours for annotation sets [126]) our approach relies on a unified algorithm that treats binary and n-ary edges on equal terms. This allows us to compute a balanced layout, and also to choose suitable representations for the binary and n-ary edges. We achieve this by assigning a bit vector $\mathbf{t} = (t_1, t_2, \dots, t_M)$ to every node $t \in V$ (the module genes) that encodes its membership in binary and n-ary edges $S_1, S_2, \dots, S_M$. That is, $t_i = 1$ if $t \in S_i$ and $t_i = 0$ if $t \notin S_i$.

To make this representation more concrete, consider the annotated module shown in Fig. 5.2. The nodes are represented as the set $V = \{$*Calm1*, *Calm2*, *Calm3*, *Kras*, *Nr3c2*, *Plcb4*$\}$. There are seven sets representing the edges and three sets representing pathway memberships. The edge sets are $S_1 = \{v_1, v_4\}$, $S_2 = \{v_1, v_6\}$, $S_3 = \{v_2, v_4\}$, $S_4 = \{v_2, v_6\}$, $S_5 = \{v_3, v_4\}$, $S_6 = \{v_3, v_6\}$, and $S_7 = \{v_4, v_5\}$. Note that nodes v_4 (*Kras*) and v_6 (*Plcb4*) have some additional outgoing edges, but their targets are not visible in the image. Therefore, we ignore these edges in this example. The pathway memberships are the *Glioma* set $S_8 = \{v_1, v_2, v_3, v_4\}$, the *Long-term potentiation* set $S_9 = \{v_1, v_2, v_3, v_4, v_6\}$, and the *GnRH signaling pathway* set $S_{10} = \{v_1, v_2, v_3, v_4, v_6\}$. Now, for example, node v_5 gets assigned the bit vector $\mathbf{t}_{v_5} = (0, 0, 0, 0, 0, 0, 1, 0, 0, 0)$ and node v_6 the bit vector $\mathbf{t}_{v_6} = (0, 1, 0, 1, 0, 1, 0, 0, 1, 1)$.

This high-dimensional representation is then used to lay out the nodes without overlap, the binary edges as curves, and the n-ary edges as contours.

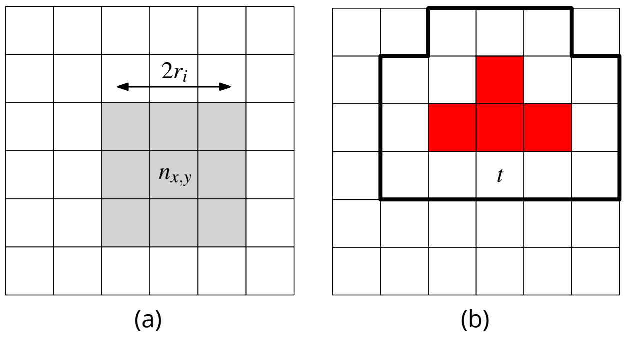


Figure 5.4: *Training neuron $n_{x,y}$: (a) The neighborhood within range r_i is trained (colored gray). (b) Certain tiles are already reserved (colored red) in the* RSOM *algorithm, item t therefore trickles outwards to the best matching free spots (outlined).*

5.2.1. Extension to Self Organizing Maps

Self Organizing Maps (*SOMs*), introduced by Kohonen [167], are artificial neural networks that are used to map high-dimensional data items to discretized low dimension. SOMs are used in a visualization setting to cluster similar items together in a 2D embedding, which results in a landscape of items based on their features [168, 169]. Typical SOMs consist of a square grid of size $N \times N$ with a neuron $n_{x,y} \in [0..1]^M$ at every grid cell. A neuron $n_{x,y}$ is a bit vector of size M which dimension matches the data items' dimensions. In our case, the data items $\mathbb{T}$ correspond to the set of nodes V in the annotated module. The training algorithm applies unsupervised reinforcement learning in an iterative fashion: at every iteration $i \in \{1, \dots, I\}$ all data items $t \in \mathbb{T}$ are considered and the neuron that matches t most closely is determined using a distance function such as the Euclidean or Manhattan norm. This neuron and its neighboring neurons within radius r_i are updated to match t even more closely by setting their respective vectors q to $q + \alpha_i(t - q)$ (see Fig. 5.4(a)). In early iterations i, the trained neighborhoods are large with r_i close to the grid size N and the training strength α_i close to 1. The parameters r_i and α_i decrease monotonically with increasing i. As such, items that differ strongly will distribute across the map to establish their own regions in the grid at early stages. Items with smaller differences are separated along the grid at a more local level as the training iterations progress.

Reservation-based training Similar items may end up at the same grid position in a standard SOM. This issue is usually solved by showing aggregate depictions of items, but we need to have separate depictions without overlap to support tasks G1-G4. Therefore, each item has to map to a unique grid position. We achieve this by altering the training algorithm:

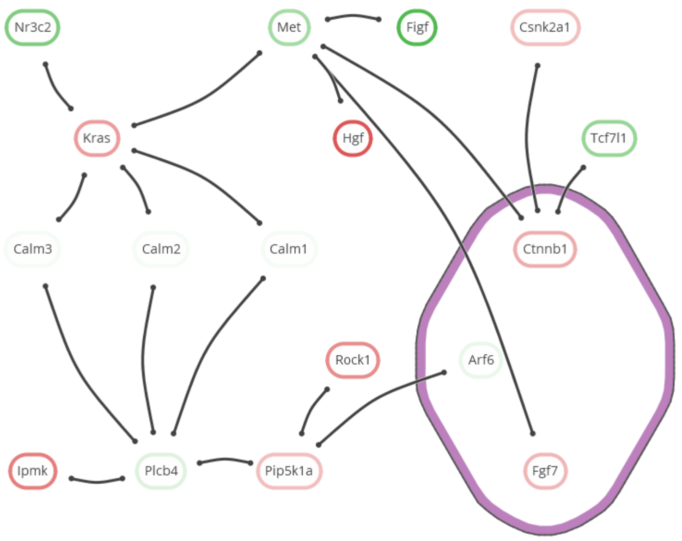
```
**Algorithm** *RSOM*($\mathbb{T}$)
1.  for $i \leftarrow 1$ to $I$
2.      do Initialize copy $\mathbb{U}$ of $\mathbb{T}$ and clear neuron reservations.
3.          while $\mathbb{U}$ contains items
4.              do Draw and remove item $t$ from $\mathbb{U}$.
5.                  Find unreserved neuron $n_{x,y}$ with smallest distance $d(t, n_{x,y})$.
6.                  Reserve $n_{x,y}$ for $t$.
7.                  for any neuron $q$ within range $r_i$ from $(x, y)$
8.                      do $q \leftarrow q + \alpha_i(t - q)$
```

The algorithm assigns items to a unique neuron after every training iteration, because, once a neuron is reserved by an item, subsequent items will ignore it. This causes a flooding effect where similar items end up in the same area of the grid and trickle outwards as the area becomes more crowded (see Fig. 5.4(b)).

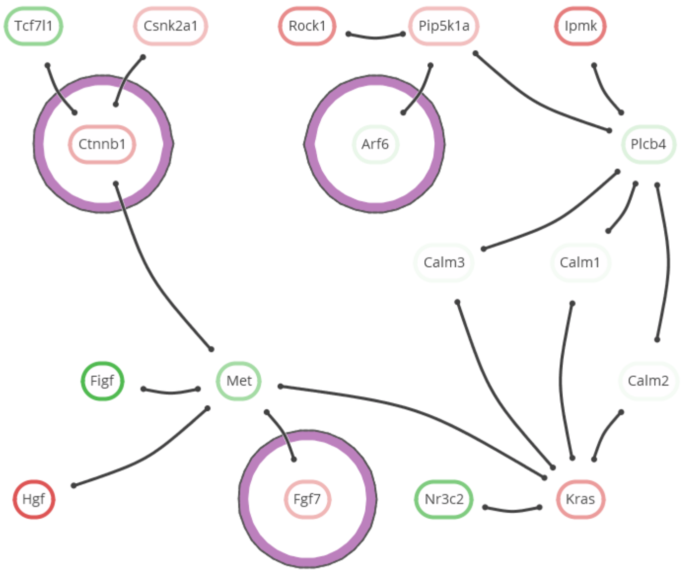
Configuration The metric distance form of cosine similarity is used as the distance function d, i.e. $d(q,p) = \cos^{-1}(q \cdot p \ / \ |q||p|)\pi^{-1}$. This measure outperforms the Euclidean and Manhattan norms in high-dimensional spaces. The SOM is trained with a learning strength and neighborhood range that decrease linearly with increasing iteration i. A standard choice is $\alpha_i = c \cdot (1 - i/I)$ and $r_i = \lfloor (1 - i/I) \cdot N \rfloor$, where $c \in (0..1)$ is a small constant that determines the initial training strength. We use $N = 2|\mathbb{T}|$ for the number of neurons and iterations, balancing node placement freedom versus required display space, and $I = 10^6/|\mathbb{T}|$ for a gradual and accurate training, respectively.

Layout preservation A new layout has to be computed whenever the user selects or deselects a set. The new layout should change little in comparison to the old layout to preserve the user's mental map. This is achieved by a simple addition to the SOM algorithm, where a new SOM is initialized with the previous configuration of the neurons, i.e., an item that was positioned at $n_{x,y}$ in the old SOM is placed at $n_{x,y}$ in the new SOM and its neighborhood is trained according to the new bit vector of the item. The new SOM retains much of the initial configuration by starting the training factor α_i at $c = 0.01$. Naturally, this imposes a trade-off between layout quality and conservation. The layout will sometimes change strongly to accommodate the addition of a set that contains many items. In contrast, the layout can be retained if only a small set that does not alter much of the topology is added. This approach does not consider a history of topological changes, as is done in online graph drawing [170] to capture temporal dynamics, but is sufficient to maintain a stable and interactive environment.

Set dominance The user is enabled to make a certain set more dominant in the layout by having the training algorithm place the items of that set closer to each other than



(a)



(b)

Figure 5.5: *Changing the dominance of a set: (a) Highly dominant set, drawing proteins of the set together. (b) Non-dominant set, where the network topology fully defines the layout.*

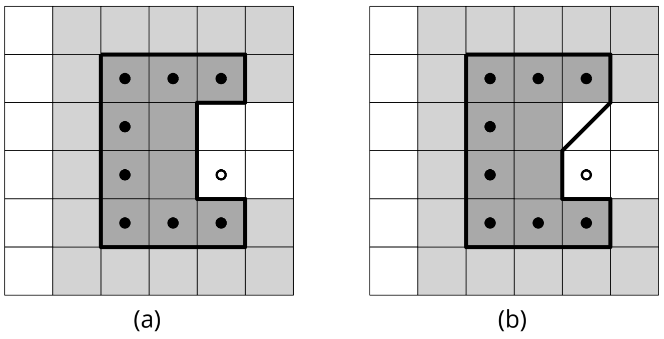


Figure 5.6: *Derivation of contours for set S_i. The darkness of a tile represents the value of the neurons' i-th component, the thick black line is the contour, dots represent items that are in S_i, and white dots are items that are not in S_i: (a) Contour that results from the union of tiles with a value above a certain threshold. (b) Refined contour with shortcuts across free tiles.*

the items of other sets. This relies on weighting the components of the item bit vectors: every S_i is given a weight w_i with $w_i = 1$ initially. The bit vectors are augmented to incorporate these weights: $t_i = w_i$ if $t \in S_i$ and $t_i = 0$ if $t \notin S_i$. The bit vector component of S_i will therefore play a more prominent role in distance metric d when the user increases w_i (see Fig. 5.5).

Assigning greater weight to a set improves the quality of its layout by coalescing its elements, which aids tasks G4 and A2. However, it also degrades the layout quality of other sets and links when their topology conflicts with the prioritized set. This stems from the difficulty of projecting elements from a high-dimensional space down to a two-dimensional space, which sometimes results in a sub-optimal layout per set. Interactive manipulation provides a way to assign different priorities to sets, and improve their layouts.

Contours The SOM's neuron grid is used to define the contours representing the active set system. Let S_i be an active set. The corresponding i-th components of the neurons define a scalar field that forms a fuzzy membership landscape for S_i. This field is similar to the density field used in Bubble Sets [126]. Now, the inclusion of the grid tile of neuron n in the contour body is determined by imposing a threshold, of for example $\frac{1}{2}$, on the i-th component (see Fig. 5.6(a)). The contour can then be tightened to reduce sharp corners by including parts of tiles that are free of items, as illustrated in Fig. 5.6(b).

After establishing the layout of the contours, we apply geometric post processing steps [212] to improve aesthetics, where all sets are legible (tasks G3 and A2) and

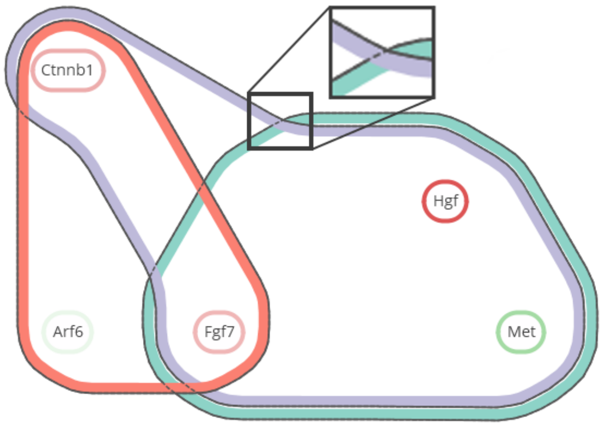


Figure 5.7: *Geometric refinement of set contours after initial layout. Corners are smoothened by dilation and erosion operations, and contours are given a thick and colored internal ribbon. Unique erosion levels create distance between contour outlines, and contour overlap is emphasized by dashed lines.*

contours form clear boundaries underneath interactions (task L). Sharp corners of the initial contours are rounded by a dilation of r, erosion of $2r$, and subsequent dilation of r (see Fig. 5.6). Here *dilate* and *erode* are equivalent to *Minkowski sum* and *Minkowski subtraction* operators with a circle of radius r [143], respectively. In addition, the contours are nested by applying different levels of erosion, enforcing a certain distance between them. The thick colored ribbons in Fig. 5.7 are obtained by taking the body b of a contour, eroding it to get a smaller body b_e, and taking the symmetric difference $b - b_e$ of b and b_e to effectively cut b_e out of b. Here, the extent of the erosions and dilations (radius r) is bounded by a fraction of the grid's tile size. This guarantees that items are contained by a contour of S_i if, and only if, these items are contained by S_i.

Set contours are drawn in descending nesting order, which is defined by their different erosion levels; the largest contour is drawn first and the smallest contour last. The contour ribbons are assigned unique colors per set and are drawn fully opaque to prevent any confusion caused by blended colors. Occlusion is mitigated by limiting the width of the ribbons. Finally, the contours are drawn a second time as dashed lines such that occluded contour sections can be inferred (see Fig. 5.7).

5.2.2. Implementation

We have implemented the technique in a Cytoscape app, and have emphasized simplicity of interaction and visual presentation in the design. The available sets are sorted by significance and listed in the *set overview* on the left, where the significance of a set is visualized as a circle, scaled logarithmically and accompanied by its scientific exponent as text (task A1). The user may select sets for inclusion in the annotated *network visualization* to the right (see Fig. 5.9(c)). All described functionalities can be used at interactive speeds for networks up to dozens of nodes, edges, and active sets, including laying out the network with the *RSOM* training algorithm. Geometric operations on the contours, such as dilations and erosions, are performed via Java Topology Suite [145].

Interaction Interactions consist of simple mouse actions. The inclusion of a set in the network visualization is toggled via the set's label in the set overview or its contour in the network visualization (task A2). Additional information about a set or node may be obtained via a hyperlink to a web page provided in the input data, enabling quick access to external information sources such as the KEGG website. This approach keeps the tool flexible, i.e., the tool itself does not have to be altered when a new kind of set or node from a different database is loaded.

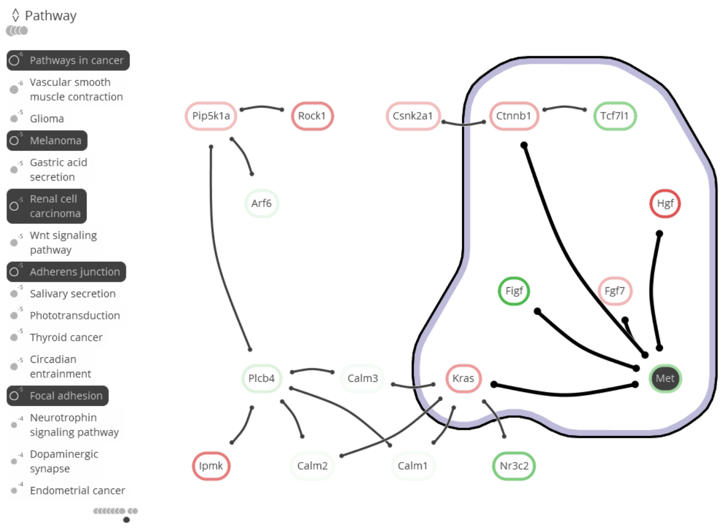
The links of a node are emphasized when it is hovered over (see Fig. 5.8(a)) such that its direct neighborhood can be discerned from its surroundings (task G2). Moreover, sets that contain the hovered node are highlighted as well. Likewise, links can be hovered to highlight their nodes and common sets. Vice versa, the contours of a set are emphasized and its comprising nodes are highlighted when it is hovered over (see Fig. 5.8(b)). This provides immediate feedback to the user about node-set relations (tasks G3 and A2) without having to select a set and consequently changing the layout of the network visualization.

The lists of annotations sets can be expanded and collapsed by clicking on their headers, and scrolled downward to sets of lower significance by turning the mouse wheel. The set circles that convey significance remain visible at all times, grouping at the list top and bottom, to guarantee the depiction of all set memberships when a node is hovered.

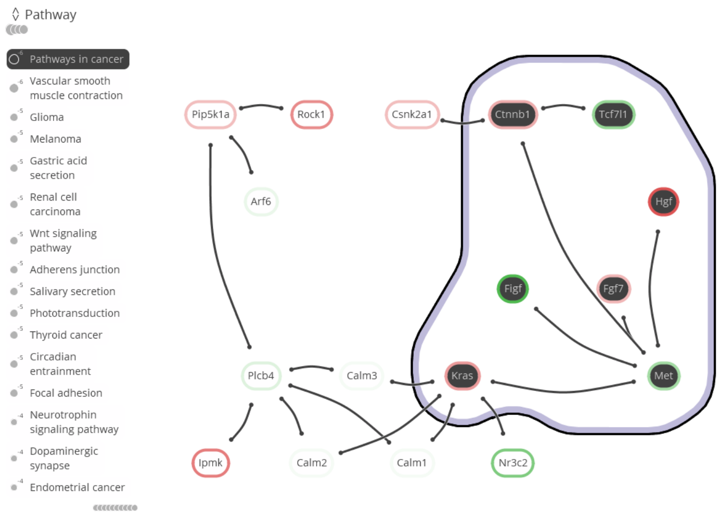
The user can adjust the dominance of a set by spinning the mouse wheel while hovering over either the set's label in the set overview or contour in the network visualization. This enables the user to give a set a central role in the layout (see Fig. 5.5(a)) or to remove any of its influence (see Fig. 5.5(b)).

All changes to the visualization caused by interaction are animated. Colors and positions of items are altered gradually. Link layout changes are animated by interpolating their control points, while contour layouts are handled by fading out the old contour and fading in the new contour. The use of layout preservation, as described previously, in combination with animations helps to preserve the user's mental map.

Color Unique, distinguishable colors are derived from Color Brewer palettes [171], and assigned to annotation sets in a cyclic manner to avoid assigning the same color consecutively. In addition, large differences in contrast are avoided. For example, text and set outlines are colored dark gray instead of black to reduce their visual dom-



(a)



(b)

Figure 5.8: *Item highlighting: (a) Hovered protein (Met) with emphasized interaction links to its neighbors on the right and emphasized sets (KEGG pathways) that contain this protein on the left. Sets outside of the list scope are grouped as markers at the top and bottom, where one set in the bottom group is emphasized. (b) Hovered set (Pathways in cancer) with emphasized member proteins, interactions, and contour.*

inance. Black is only used when items are hovered over or highlighted such that they attract attention, as shown in Fig. 5.8. Moreover, labels of selected sets (in the set overview) are colored black to ensure that they are readable in a colored surrounding. Node labels have a white background to make sure that their text is legible when drawn on top of a set ribbon with a dark color. Likewise, links have halos that make them easier to distinguish and their intersections more pronounced.

Cytoscape integration eXamine is tightly integrated into Cytoscape. Cytoscape's group functionality is used to represent sets and we rely on the table import functionality for importing both the set and node annotations. The user is also able to group sets into different categories. The Cytoscape node fill color map attribute is used to color the nodes in eXamine according to gene expression score (task G1). The user therefore has the freedom to define the desired color map via Cytoscape. The user can invoke eXamine on the currently selected nodes via the eXamine control panel. There the user can select which categories to show as well as the number of sets per category. In addition, the user can specify that the Cytoscape selection should be updated to match the union or intersection of the selected sets in eXamine (see Fig. 5.9). This enables the use of eXamine with any kind of module extraction algorithm and/or filter method in Cytoscape, which includes manual node selection.

5.3. Case study of US28-mediated signaling

We demonstrate how a domain expert can use eXamine by working out a case study in which a data set is re-analyzed (this work was done by several biologist co-authors of the paper associated with this chapter). While this data set has been studied extensively, it was possible to derive a new hypothesis via eXamine.

The *Human Cytomegalovirus (HCMV)* is a highly-contagious herpes virus [172]. Infection with HCMV in healthy humans usually does not result in symptoms. However, in humans with a compromised immune system the virus is correlated with diseases such as hepatitis and retinitis [173]. In addition, HCMV gene products have been detected in various tumors even though HCMV is not considered to be an oncogenic virus. Experts therefore hypothesize that the virus may act as a stimulating factor during onset and development of cancer without being a root cause [174-176].

HCMV is responsible for the production of several viral G protein-coupled receptors (vGPCRs). Of these vGPCRs, US28 is the most studied and is characterized as chemokine sink [177]. Chemokines are signaling proteins that induce cell migration. Moreover, US28 hijacks the host cell's signaling pathways, stimulates proliferative signaling pathways [178-182]. Previous studies focused on transcriptome analysis to evaluate pathways that are affected by US28. Differentially expressed genes involved in HCMV-induced disease symptoms were identified and related to known pathways [180, 181]. However, this analysis did not include network-based module extraction and enrichment.

To identify additional deregulated signaling due to US28, we analyzed the same data overlaid on the KEGG mouse network [155]. The network consisted of 3863 nodes and 29293 edges. Gene p-values, reflecting whether genes are significantly differentially expressed, were derived using RMA [183] and LIMMA [184]. Heinz [154], a tool for identifying differentially expressed modules, was then applied using a false discovery rate of 0.0007. This resulted in a module of 17 proteins. Finally, enrichment analysis using TopGO [185] was performed to annotate this module with enriched GO-terms and KEGG pathways (see Fig. 5.9).

These data processing steps correspond to the initial steps in Fig. 5.1. The subsequent analysis of the annotated module aims at obtaining new insights about US28-mediated signaling. The analysis follows the visual analytics cycle consisting of *observation*, *knowledge*, *questions* and *exploration*, finalized by a hypothesis.

C1 Two familiar pathways

Observation The KEGG pathway annotation sets show significant presence of *Pathways in cancer* and *Phosphatidylinositol signaling* (with p-values of $5.6 \cdot 10^{-6}$ and $1.0 \cdot 10^{-6}$, respectively).

Knowledge An oncomodulatory role has been proposed for US28 [174-176], which coincides with the presence of *Pathways in cancer* and makes the genes annotated by this term of interest. *Phosphatidylinositol signaling* corresponds to previous work linking US28 to Phosphatidylinositol-mediated calcium responses [178, 186].

Question Which parts of the module are involved in *Pathways in cancer* and *Phosphatidylinositol signaling*?

Interaction Tag the *Pathways in cancer* and *Phosphatidylinositol signaling* annotation sets (see Fig. 5.9(a)).

C2 Choosing sides

Observation Clear division of the module is apparent after tagging the two familiar pathways. Genes *Arf6*, *Csnk2a1*, *Csnk2a1*, *Ipmk*, *Nr3c2* and *Rock1* are not part of the pathways but have direct, unambiguous interactions with either of the pathways.

Knowledge Because of the known involvement of US28 in *Phosphatidylinositol signaling*, we do not focus on the genes of this pathway (*Calm1..3*, *Plcb4*, *Pip5k1a*), nor on the directly interacting genes (*Arf6*, *Ipmk*, *Rock1*). Instead, the *Pathways in cancer* genes *Kras*, *Met*, *Figf*, *Hgf*, *Fgf7*, *Ctnnb1* and *Tcf7l1*, and directly interacting genes *Nr3c2* and *Csnk2a1* may lead to new insights in US28-mediated signaling and ultimately the oncomodulatory role of HCMV.

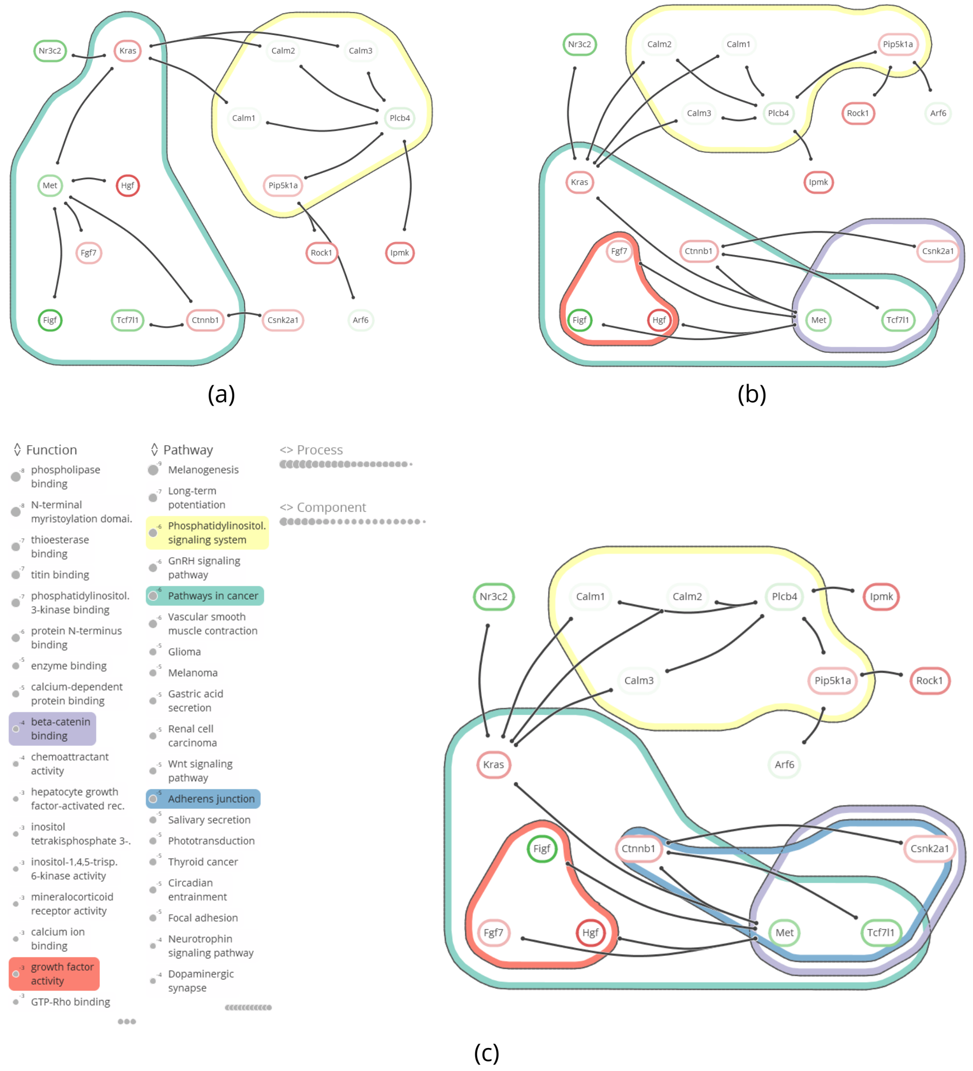


Figure 5.9: *Case study snapshots. Gene differential expression is shown as a colored box drawn around the node label (green for under-expression and violet for over-expression): (a) The annotated module after tagging of the two familiar pathways* Pathways in cancer *and* Phosphatidylinositol signaling system *in C1. (b) The annotated module after tagging functions* Beta-catenin binding *and* Growth factor activity *in C3 and C4. (c) The fully annotated module, including annotation set overview, from which the hypothesis of C5 is derived.*

Question Do any of the aforementioned genes in or adjacent to *Pathways in cancer* lead to new insights in US28-mediated signaling?

Interaction Hover over the genes in and close to *Pathways in cancer* to determine mechanisms of interest.

C3 A twist of β-catenin

Observation The genes in *Pathways in cancer* can be divided roughly into two subsets: those that are annotated by *growth-factor activity* and those annotated by *β-catenin binding* (see Fig. 5.9(b)). *Csnk2a1*, *Tcf7l1* and *Met* are part of the latter annotation set, where *Tcf7l1* and *Csnk2a1* are down- and up-regulated, respectively. Expression of the neighboring *Ctnnb1* (β-catenin) is up-regulated.

Knowledge β-catenin signaling results in elevated protein levels of the TCF/LEF transcription factor family that contains the protein encoded by *Tcf7l1*. Although *Tcf7l1* is down-regulated, a recent study shows that this is not reflected at the protein level and that US28 induces β-catenin signaling [182]. In the same study, involvement of WNT/Frizzled via the canonical signaling pathway was ruled out and a hypothesis stating that US28-mediated signaling of β-catenin proceeds via ROCK1, which is also present in the module, was postulated.

Question Are there alternative mechanisms explaining the activation of β-catenin?

Interaction Tag the *Growth factor activity* annotation set (see Fig. 5.9(b)).

C4 Growing knowledge

Observation *Fgf*, *Hgf* and *Figf* are annotated with *Growth factor activity* and connected to β-catenin via *Met*.

Knowledge MET is a receptor tyrosine kinase, whose only ligand is HGF. Therefore we can rule out the links from *Met* to *Fgf* and to *Figf*. In fact, these links are artifacts of how the mouse network was constructed from KEGG pathways. These artifacts often link whole groups of genes such as, in this case, growth factors to receptor tyrosine kinases.

Question Does the *Hgf* to *Met* axis relate to β-catenin activation?

Interaction Hover over *Met* and *Ctnnb1* (β-catenin).

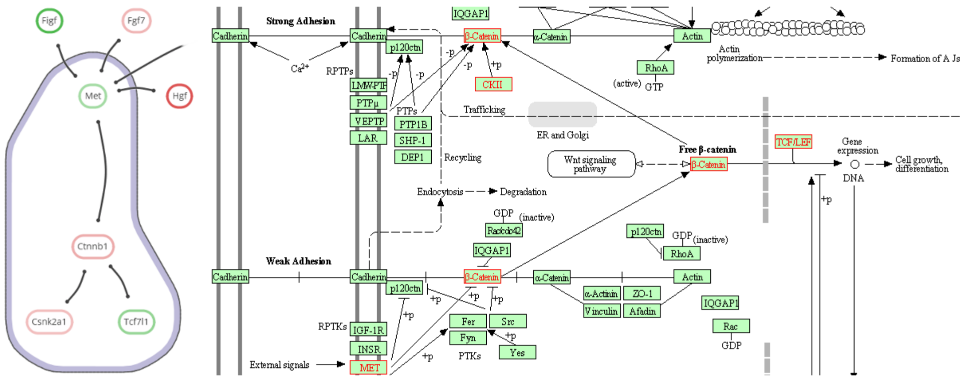


Figure 5.10: *Connection between Met and β-catenin. Proteins that are associated to the selected* Adherens junction *at the left and corresponding KEGG pathway information at the right, where reactions catalyzed by module proteins are marked in red. Activation of MET by its ligand HGF results in the phosphorylation of β-catenin. This in turn results in its release from cadherin-complexes on the cell membrane into the cytoplasm.*

C5 New insights

Observation *Met* and *β*-catenin are both part of the *Adherens junction* pathway, as are *Tcf7l1* and *Csnk2a1* (see Fig. 5.9(c)).

Knowledge Adherens junctions bind two cells together, keeping multiple cells in place. Alternative mechanisms have been described that explain *β*-catenin activation via the release of *β*-catenin from cell to cell adherens junctions (e.g. [187]). US28 promotes cell migration [188, 189], which causes the loss of cell to cell contacts with subsequent release of *β*-catenin into the cytoplasm. This may explain increased levels of *β*-catenin as found previously [182].

By requesting additional information for *Adherens junction* via eXamine, showing an external website by KEGG, we find an indirect connection between *Met* and *β*-catenin in the pathway (see Fig. 5.10). Activation of MET via HGF mediates the release of *β*-catenin from adherens junctions, resulting in increased TCF/LEF levels [190, 191].

Hypothesis Combining this with the growth factor observations of C4 leads to the following hypothesis:

- US28-mediated up-regulation of *Hgf* results in elevated levels of the corresponding HGF protein;
- The subsequent activation of MET results in the release of *β*-catenin into the cytoplasm;

- Subsequent translocation into the nucleus leads to enhanced TCF/LEF activation.

Synopsis

The hypothesis is being validated experimentally. Preliminary results indicate that the up-regulation of *Hgf* is indeed reflected at the protein level. Should this hypothesis turn out to be true, we would obtain crucial insights into one of the mechanisms by which the HCMV-encoded chemokine receptor US28 rewires cellular signaling. Ultimately, we would like to understand how this virus achieves its oncomodulatory role and how this can be disrupted.

5.4. Discussion

The analysis tasks described in the background section guided the design decisions that we have taken in the implementation of eXamine. These decisions are motivated via the analysis cycles of the US28 case study.

Overview The benefit of a spacious annotation set overview follows from the first cycle (C1), in which the categorized, ranked, and legible annotation lists enable the fast recognition of two familiar and significantly represented pathways (task A1). Subsequent tagging of the two pathways reveals their module genes (task A2) and concisely drawn contours emphasize the division of the module into two parts and some additional genes that are not part of the pathways.

An annotation table, separate of the network, would not have made this division as apparent. The main reason is that annotation set transitions along gene interactions are not explicit in such a representation. In contrast, such cross-contour interaction links are clearly visible in eXamine (e.g. the transition from *Kras* in *Pathways in cancer* to *Nr3c2* outside of *Pathways in cancer*).

Annotated genes The need to focus on specific genes and their properties appears in the second analysis cycle (C2), in which genes of *Pathways in cancer* are inspected for annotations of interest (task G3). Highlighting annotations by hovering over genes enables fast identification of relevant annotations in the stable overview that oriented the analyst in C1. Vice versa, hovering an annotation of interest (*β-Catenin binding*) confirms that it is shared by *Csnk2a1*, *Tcf7l1*, and *Met* (task G4). The same observations could have been made from an annotation table. However, the topological characteristics of these three genes would have been harder to discern, i.e., their direct interaction with *Ctnnb1* (task G2). This also applies to other set visualizations without depiction of network topology, such as Venn or Euler diagrams, as shown in Fig. 5.3(a) and (b). To make the topology of the gene interactions more explicit, a node-link visualization could be used. For example, Fig. 5.3(c) shows the module laid out by one of the built-in force-directed layout algorithms of Cytoscape with all five annotation sets superimposed as BubbleSets. However, the structure of the annotation sets is hard to discern, and it is not immediately clear that nodes belonging to the β-catenin binding set (blue shape) form a proper subset of the Adherens junction set (yellow shape).

Integration The third cycle (C3) shows the importance of gene expression values (task G1), which is not limited to the interpretation of genes in isolation but along multiple genes, their interactions, and shared annotation sets. The importance of integrated support for all analysis tasks follows from the remaining cycles (C4-C5), where multiple deductions are made in succession via multiple tasks. Here, tagging relevant pathways enables the analyst to build up a context for making deductions.

Limitations eXamine is designed to accurately convey small and annotated modules, consisting of up to about thirty proteins and categories of up to about twenty annotations (note that these limits are not hard). The case study shows that common analysis tasks for these modules are covered. Scalability is a concern as our approach focuses on small modules to enable accurate depiction of sets contours; it is not possible to construct a comprehensive layout if the module consists of hundreds of proteins or if there are dozens of annotation sets to visualize at the same time. Both aspects make visual analysis ineffective. This is a natural limitation of any visualization approach based on node-link diagrams and set contours, however.

Our technique relies on a focus and context approach, in which the network and set system has been pruned down to the most relevant components first. Communicating small-scale information is given priority to support hypothesis generation at the level of individual proteins and their interactions, as follows from the targeted analysis tasks. Nonetheless, the tool is capable of visualizing modules of up to a hundred proteins, albeit with less legibility of interactions and annotations.

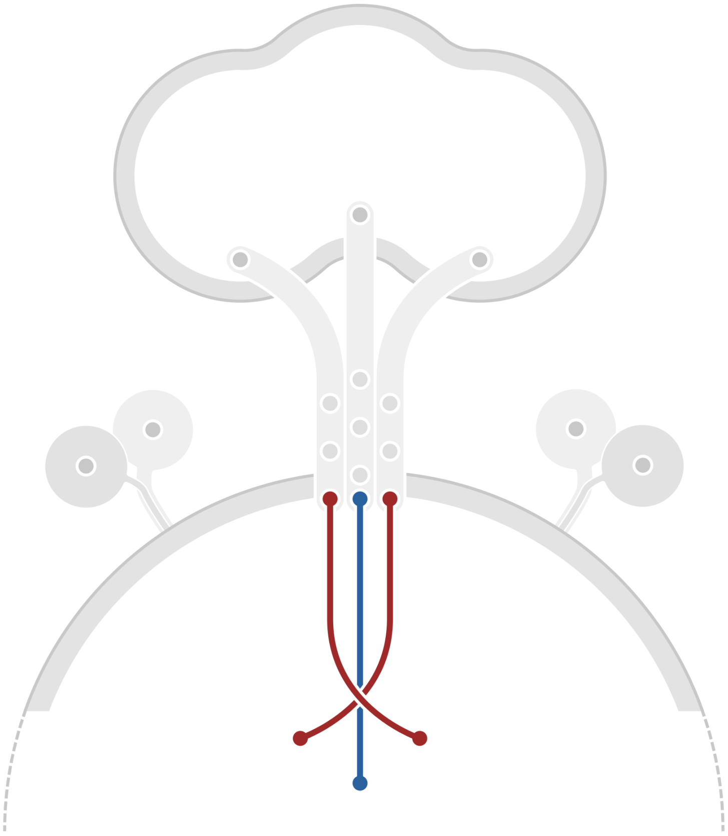
The integration of eXamine into Cytoscape mitigates many scalability issues. Cytoscape, for example, provides a global view of the network, in which the user can zoom in on smaller subnetworks for more in-depth analysis by eXamine. In addition, the integration into Cytoscape provides access to further analysis algorithms.

The extended SOM algorithm embeds an annotated module to reflect its topology, i.e., the distances between its proteins based on common interactions and annotations. This does not guarantee optimal aesthetics however, and unnecessary link and contour intersections can sometimes occur. The analysis tasks targeted by eXamine are not much hampered by such intersections since all interactions, annotations, and their interplay remain pronounced. However, to communicate analysis results, aesthetics might need further improvement. This could be done along the same lines as Kelp Diagrams (see Chapter 4), by weighing aesthetic criteria, such as the number of intersections and shape complexity, against each other and formulating this as an optimization problem. The associated algorithms [16] are complex, and it is difficult to integrate these into an interactive system.

5.5. Conclusions

We have proposed a visualization approach that enables the analysis of small and annotated network modules, and have implemented this in the Cytoscape app eXamine. Our approach displays sets as contours on top of a node-link layout. We have introduced an extension to the self-organizing maps algorithm to lay out module edges and annotation sets in a unified way. The added value of our approach has been demonstrated in a case study of a US28-mediated signaling module, in which a novel hypothesis about the way US28 induces β-catenin signaling has been derived.

6
Dual Adjacency Matrix



K. Dinkla, N. Henry Riche, and M.A. Westenberg

Parts of this chapter have been published in Computer Graphics Forum [214].

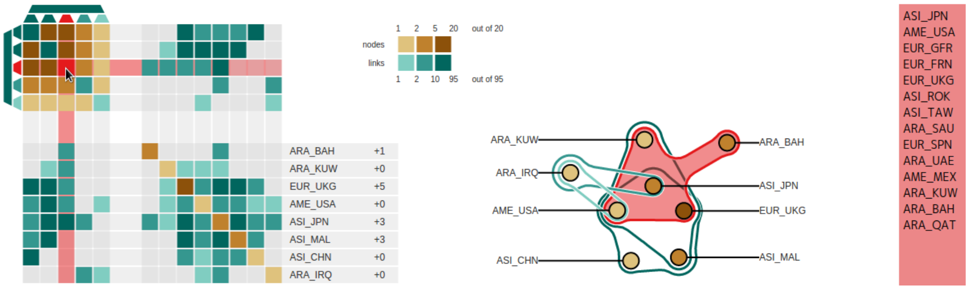


Figure 6.1: *Dual adjacency matrix (left), node-link-contour diagram (middle), and highlighted group nodes (right) that depict a trade network of countries (nodes) with substantial changes in volume traded (links) over a fifty year period. One group of links is selected in the top-left matrix, which covers countries such as Japan (ASI_JPN) and the USA (AME_USA).*

Node grouping is a common way of adding structure and information to networks that aids their interpretation. However, certain networks benefit from the grouping of links instead of nodes. Link communities, for example, are a form of link groups that describe high-quality overlapping node communities. There is a conceptual gap between node groups and link groups that poses an interesting visualization challenge. We introduce the Dual Adjacency Matrix to bridge this gap. This matrix combines node and link group techniques via a generalization that also enables it to be coordinated with a node-link-contour diagram. These methods have been implemented in a prototype that we evaluated with an information scientist and neuroscientist via interviews and prototype walk-throughs. We demonstrate this prototype with the analysis of a trade network and an fMRI correlation network.

6.1. Introduction

Many networks are derived through experimental observation of real-world systems for analysis purposes. Social networks, for example, describe interactions between people and provide insights about the functioning of society (see Fig. 6.2(a)). Some of these networks are dense, in which most nodes are so interconnected that their individual roles are of less interest than the concert of their interactions. This phenomenon appears as the notion of a network module, or *community*, which is a dense network section (with a high link-to-node ratio) that reflects part of a system that is likely to have a special role (see Fig. 6.2(b)). For example, communities in the social network of a company could be correlated to departmentalization, where team members are likely to interact. Likewise, communities emerge via natural selection in organisms and appear in protein interaction and metabolic networks [51, 64]. Communities are often defined in terms of node groups.

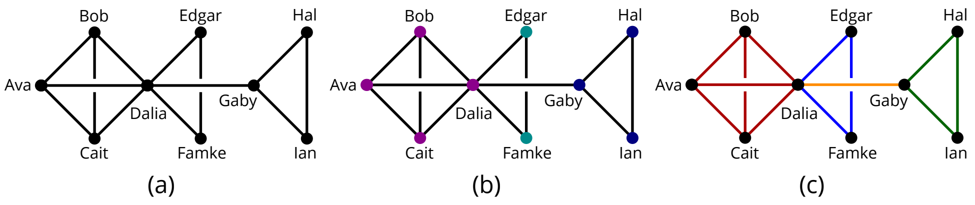


Figure 6.2: *Example of a network and derived community structures: (a) Plain social network, people are nodes (dots) and their interactions are links (connecting lines). (b) Densely interconnected nodes of (a) have been grouped into communities, in which* Dalia *is part of a single community in spite of her widespread interactions. (c) Densely interconnected links of (a) have been grouped into communities, in which* Dalia *is part of multiple communities.*

Node group A set (or cluster) of nodes that together fulfill a role within a network. Node groups are disjoint.

For visualization purposes, node groups ease the aggregation of networks into several joint nodes for which information is summarized. However, certain networks benefit from grouping (or clustering) links instead of nodes.

Link group A set (or cluster) of links that together fulfill a role within a network. Link groups are disjoint.

Suppose the links of a network are accompanied by a time series. Grouping links by similar behavior over time will expose links (and the nodes that they connect) that act in concert, indicating a shared role. Likewise, link groups that are clustered by network connectivity can be used to determine high-quality overlapping node communities [192], as shown in Fig. 6.2(c).

Link groups are more difficult to grasp as a concept than node groups and therefore pose an interesting visualization challenge. We bridge this conceptual gap by contributing:

- Generalization and combination of node and link group techniques into a Dual Adjacency Matrix (DAM) and node-link-contour diagram;
- A prototype implementation and demonstration of our approach on a trade network and fMRI correlation network;
- Informal evaluation by an information scientist and neuroscientist via interviews and prototype walk-throughs.

6.2. Related Work

Visualization of network topology has been the subject of much research [5, 6], where node groups often occur to, for example, support node time-series [193] or multivariate [194] analysis. Shifting focus from nodes to links has already appeared in various forms. Bundling links based on the position of their nodes in a predefined hierarchy reveals correlations between links based on the properties of their nodes [22]. Visual manipulation of network topology can be avoided via explicit visualization of link to link relations by introducing an extra type of link [195].

An overview of node community visualization can be found in [196]. Many approaches involve node-link diagrams in which community memberships are visualized by layout [196] and color [197, 198]. A dual, community-centric approach is taken in [197], where communities are depicted as nodes and their overlaps as weighted links. These techniques have all proved effective, either for arbitrary overlapping communities or those that result from specific detection algorithms. However, to the best of our knowledge, no visualization techniques have been explored for link groups and the node groups that they induce.

Visualizing overlapping node groups while abstracting from the underlying network topology is equivalent to the visualization of a set system or undirected hyper graph. Venn and Euler diagrams represent these set systems as overlapping shapes with elements placed in the shapes according to their set memberships. Here the layout of shapes and elements plays an important role [134-136] and this layout is sometimes constrained as well [126, 147]. Some methods give priority to visualizing the distribution of elements among sets, instead of set system topology [199-201]. Various matrix-like representations exist as well [202, 203]. An overview of set system visualization can be found in [204].

Table 6.1: *Analysis tasks from Node-centric and Link-centric perspectives.*

	Node-centric	*Link-centric*
T1. Membership	Find link groups that cover given nodes	Find the nodes of a given link group
T2. Overlap	Find link groups that share given nodes	Find nodes shared by given link groups
T3. Path	Find paths between nodes via link groups	Find paths between link groups via nodes
T4. Cluster (clique)	Find multiple nodes that share many link groups	Find multiple link groups that share many nodes
T5. Component	Find (dis-)connected components of nodes	Find (dis-)connected components of link groups
T6. Hub	Find a node that is covered by many link groups	Find a link group that covers many nodes
T7. Bridge	Find nodes that are sole link group connectors	Find link groups that are sole node connectors
A. Attribute	Compare attributes of links that cover given nodes	Compare attributes of given link groups

6.3. Link Group Analysis Tasks

Related research [9, 136, 205] has formalized a list of important tasks performed by analysts on networks, set systems, and node groups in networks. Network tasks capture how nodes are related to each other via their *link connectivity*. As a dual to this, we consider relating links to each other via their imposed groups as well as *node connectivity*, and identify a set of analogous tasks, which capture how link groups are related to each other via shared nodes. Additional link attributes, possibly used to derive link groups, are included in these tasks (see Table 6.1).

6.4. Concept

To the best of our knowledge, visual aggregation and navigation techniques have so far only focused on node groups. We transfer established techniques from node groups to link groups and introduce visualizations that couple node and link group perspectives.

Dual Adjacency Matrix Both node and link perspectives are not only combined but unified in the dual adjacency matrix, which consists of four quadrants, as shown in Fig. 6.3.

The bottom-right quadrant (see Fig. 6.3(b)) is the familiar matrix that shows adjacencies between node groups. Node groups (along the diagonal) are colored brown according to their size and the number of links that connect two node groups is shown with a green color scale (see Fig. 6.3(e)). For example, a node group in Fig. 6.3 is highlighted in blue, which includes its matrix row and column. This node group consists of *Ava* and two additional people (indicated by a *+2*), and it is connected to the node group of *Dalia*.

The top-left quadrant is the dual of the matrix at the bottom-right; it shows adjacencies between links or link groups. Again, link groups (along the diagonal) are colored green according to their size, and the number of nodes shared between two link groups is color coded in brown. For example, one link group in Fig. 6.3(a) is highlighted in red and shares nodes with two other link groups.

The bottom-left and top-right quadrants are symmetric and connect the top-left link groups to the bottom-right node groups, showing which node groups are covered by which link groups. For example, the bottom-left and top-right quadrant tiles that are highlighted in black in Fig. 6.3(c) show the connection between the blue and red node and link groups respectively. Here, the green color coding shows the number of links that cover nodes from a node group.

Node-link diagram Matrix visualizations are suited for the visualization of dense networks, where link group overlaps are common. In case of simple link group topology, we also show how to create node-link diagrams. Coordinating these diagrams

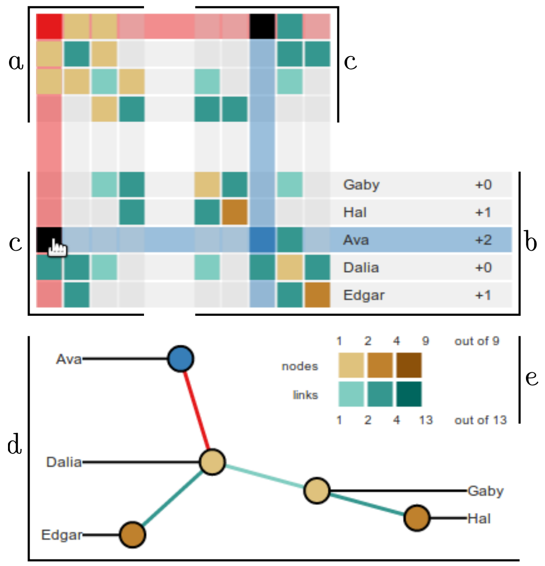


Figure 6.3: *Concept of the dual adjacency matrix, in which the network of Fig. 6.2(c) is depicted while aggregated according to its link groups: (a) The rows and columns of the top-left quadrant represent link groups in which the diagonal shows the number of links in each link group, and the quadrant remainder shows the number of nodes that connect the row and column link groups. (b) The rows and columns of the bottom-right quadrant represent node groups, in which the diagonal shows the number of nodes in each node group, and the quadrant remainder shows the number of links that connect row and column node groups. (c) The rows and columns of (a) and (b) extend into the two remaining quadrants that show which node groups are covered by which link groups. (d) Node-link diagrams that match the dual matrix are used to depict link and node group topology when it is sparse. (e) Numbers of nodes and links are encoded by color scales, and selected groups are shown in red and blue.*

with a dual adjacency matrix via interactive highlighting eases the transition from reading a familiar node-link diagram to reading a more complex matrix. For example, the node-link diagram of Fig. 6.3(d) shows the node group of *Ava* highlighted in blue and covered by a link group highlighted in red. It also shows that the red link group covers the *Dalia* node group, as can be seen in Fig. 6.3(c).

6.5. Network Aggregation

We regard node and link groups as each other's dual, where nodes and links can be interchanged. The dual of a regular network is also known as a *link-to-node dual* or *line graph*.

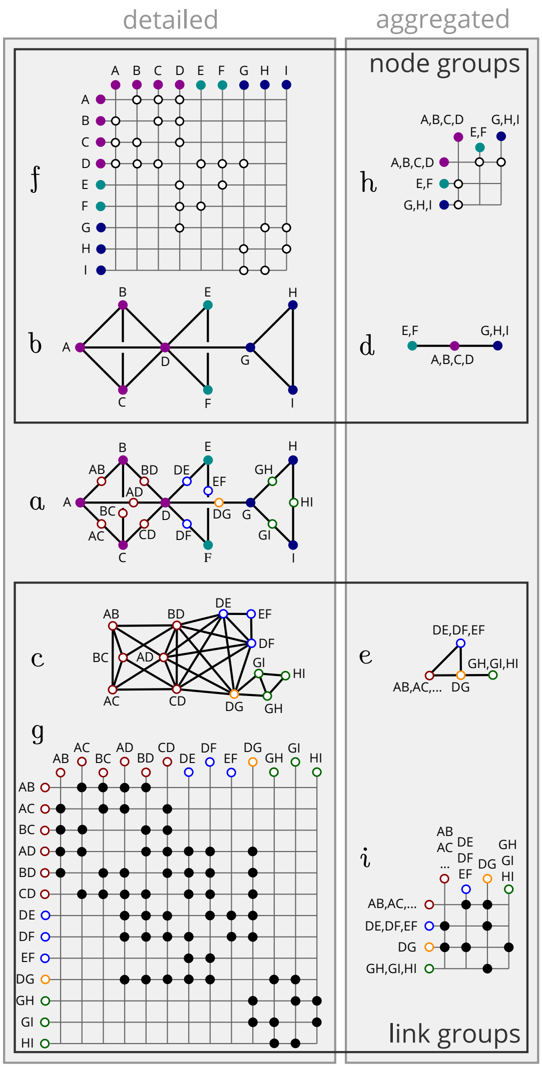


Figure 6.4: *Overview of network aggregation applied to the network of Fig. 6.2: (a) Bipartite network that bridges the node and link duality of (b) and (c), in which the node-to-node (solid dots) and link-to-link (hollow dots) connections correspond to node-link-node and link-node-link paths respectively. (b) Node-link diagram in which nodes (solid dots) are colored by group. (c) Node-link diagram of the dual of (b) in which links (hollow dots) are colored by group. (d) and (e) Node-link diagrams in which the respective groups of (b) and (c) are aggregated into single nodes. (f),(g),(h), and (i) Adjacency matrices of the (b),(c),(d), and (e) networks respectively.*

Node groups Node groups are derived from node attributes [121, 206] or topology. For example, nodes can be grouped by similar attribute values, short topological distance, or neighborhood similarity. The node groups in Fig. 6.4(b), indicated by color, induce the aggregated network of Fig. 6.4(d). Every node in the aggregated network represents a group of people and every link represents the presence of an interaction between one or more members of the two groups. Both networks can also be represented as *node adjacency matrices*, shown in Fig. 6.4(f) and (h).

Link groups The grouping of links can also be expressed as a grouping of nodes in the dual of Fig. 6.4(b). This dual can be represented by either a node-link diagram or a link adjacency matrix as shown in Fig. 6.4(c) and (g) respectively. The aggregated adjacency matrix of Fig. 6.4(i) depicts every link group as a row and column in the matrix (with accompanying labels to the sides), and shared nodes as the dots at intersections.

The conceptual gap between grouping nodes and grouping links can be closed via a bipartite graph interpretation (see Fig. 6.4(a)). Nodes and links are shown as solid and hollow dots that are connected if the associated nodes and links are connected in the original network. This interpretation structures the node-centric versus the link-centric tasks of Section 6.3, for which an observer traces connections between the two groups of the bipartite graph and the only difference between the task categories is the type of node that the observer starts from.

6.6. Construction of a Dual Adjacency Matrix

The link adjacency matrices of Fig. 6.4(g) and (i) display overlaps of only two link groups at a time, while there is a need to oversee the intersection of an arbitrary number of groups (tasks T2 and T4). For example, D in Fig. 6.4(a) is covered by three link groups, but this is difficult to see in Fig. 6.4(g). We therefore extend the link adjacency matrix into a *Dual Adjacency Matrix (DAM)*, where link groups (Fig. 6.5(a)) are regarded in terms of their overlap combinations (Fig. 6.5(b)) to construct additional matrix quadrants (Fig. 6.5(c) and (d)) that translate link groups into node groups (Fig. 6.5(e)).

Link group intersections The need to oversee intersections of arbitrary numbers of link groups can be met by grouping nodes according to the link groups that cover them, as shown in Fig. 6.5(a) & (b). This is similar to visualizing overlapping sets by grouping nodes according to set membership combinations [203, 204]. However, in this case each set consists of the nodes covered by a link group. The resulting link group to node group relationships can also be shown as the matrix in Fig. 6.5(c), where node groups are not predefined (as in Fig. 6.4) but derive from link groups. These node groups also derive the node adjacency matrix of Fig. 6.5(d), in which row and column intersections depict links that are shared between node groups. This type of adjacency matrix enables node-centric hub (T6) and bridge (T7) identification.

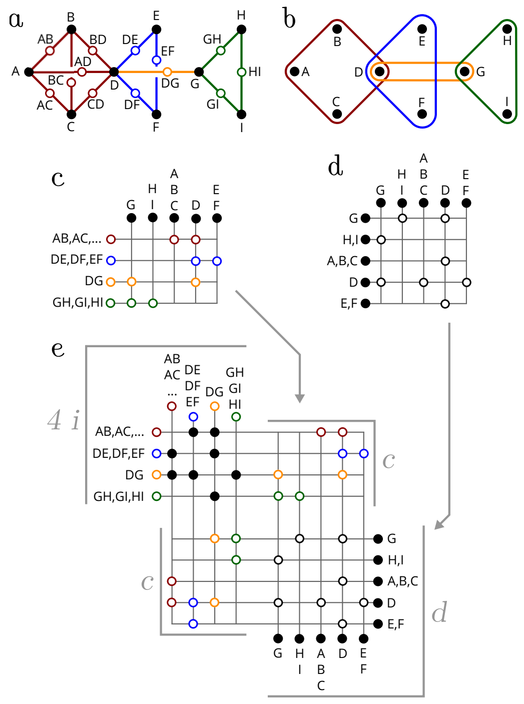


Figure 6.5: *Providing a node group interpretation of link groups with a dual adjacency matrix: (a) Network of Fig. 6.4(a), but with links colored to emphasize their groups. (b) Euler diagram of the set system that is induced by the link groups of (a), in which every set contains those nodes covered by its corresponding link group. (c) Set membership table (or matrix) of the set system of (b) that depicts all link group overlaps as the composition of node groups. (d) Node adjacency matrix of the node groups of (c). (e) Combination of (c), (d), and Fig. 6.4(i) that forms a dual adjacency matrix.*

Combination and extension The matrix of Fig. 6.5(c) forms the bridge between node and link groups that enables combination of the node and link adjacency matrices into the dual adjacency matrix shown in Fig. 6.5(e). This combination supports both node- and link-centric tasks, in particular membership (T1) and overlap (T2).

The top-left quadrant of a dual adjacency matrix shows the nodes that connect link groups and the bottom-right quadrant shows the links that connect node groups. This leaves the quadrants of Fig. 6.5(c) open to what their cells (row and column intersections) represent; either connecting nodes or connecting links. Showing connecting nodes is superfluous, because our link groups induce node groups such that any cover of a node group by a link group is complete (all nodes are covered). Representing links does provide additional information however, as shown in Fig. 6.3(c). It conveys which links of a link group cover a node group; a column shows how a link group decomposes over node groups, and a row shows how the neighborhood links of a node group divide over link groups.

6.7. Construction of a Node-link Diagram

The dual adjacency matrix focuses on the visualization of link groups and their overlaps (tasks T2, T4, and T6) as an extension of common adjacency matrices. Adjacency matrices are difficult to read and they perform poorly on global topology tasks (T3, T5, and T7) in comparison to node-link diagrams [108]. We have therefore also explored link group visualization as forms of node-link diagram.

Detailed node-link diagram Early prototypes featured a node-link diagram of the entire network, in which nodes have a pre-computed position (see Fig. 6.6(a)). Interactions with the link and node groups of the duality matrix are coordinated with this node-link diagram; nodes and links are colored according to hovered groups. This approach was valued by users for small and sparse link groups but did not scale due to a lack of aggregation.

Aggregated node-link diagram Aggregating the detailed node-link diagram down to the groups of the dual adjacency matrix gives a less cluttered but more abstract visualization: Every node group appears as a node and every link group as multiple lines; one line runs between two nodes for every link group that covers them. This approach is common for the visualization of multiple (overlapping) link types, but in this case it shows link groups.

Node-link-contour diagram Link groups facilitate the creation of the overlapping shapes of an Euler diagram because their individual links can be inflated and combined (see Fig. 6.6(c)). The links that underlie a community therefore act as a skeleton, which can be inflated to form hulls [147, 207]: a contour is derived per link group by dilating its links (the application of a Minkowski sum [143] with a circle), dilating covered node groups with a greater radius for emphasis, then eroding the contour to smoothen it, and finally subtracting the areas around non-member nodes to avoid invalid overlaps. Contours are separated by dilating with different radii at nodes that are shared by multiple link groups (see the nested contours in Fig. 6.6(c)). Contours are also cohesive per link group, provided that the link groups themselves are cohesive (which is often the case [192]). To provide more insight into network topology it is also possible to visualize aggregated links at the expense of some additional clutter (see Fig. 6.6(d)).

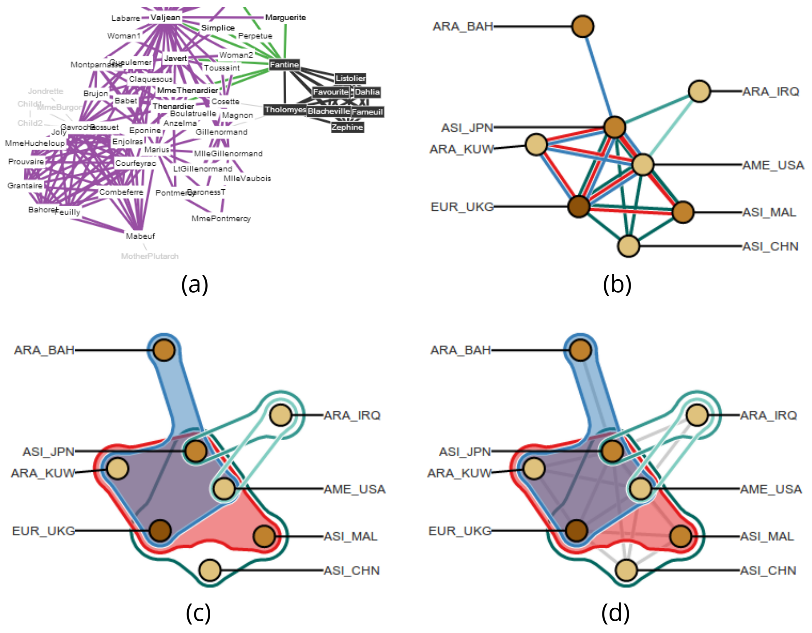


Figure 6.6: *Node-link diagrams that incorporate link groups: (a) Detailed, full node-link diagram with color coding of nodes and links via interaction. (b) Aggregated node-link diagram with links between node group pairs, color coded like (a). (c) Link group contours derived (and color coded) from the links of (b). (d) Addition of links between group pairs to (c) for improved depiction of topology.*

6.8. Prototype Design

We implemented the DAM in a prototype that was refined in an iterative fashion with feedback from link group experts. The prototype includes additional node and link information.

Node labels Node groups are labeled to provide some indication of their contents. This label consists of the name of the most important node in the group (by input score) and is shown at an extension of its matrix row (see Fig. 6.3(b)). It also shows the remaining number of nodes in the group to emphasize it being a group.

Multi-level link groups We expect input link groups to be the result of clustering, based on link topology and/or attributes like a time series. Such clusterings are often hierarchic, which is why the prototype supports a link hierarchy. This hierarchy is mirrored at the top and side of the top-left matrix quadrant as icicle plots that extend to the rows and columns of the matrix (see Fig. 6.1). The sides of this icicle plot are tapered to get a tree-like representation with pronounced hierarchy branches. Interactive hierarchy navigation is enabled for better scalability, where changes to the visualizations due to splitting high-level groups are animated.

Aesthetics The colors for highlighted groups are derived from Color Brewer [171]. Strong contrast and hard outlines are avoided, though black is used for highlights and node legibility. Matrix rows and columns are of a translucent gray such that their intersections are more pronounced. A large space is placed between the dual matrix quadrants to make them appear cohesive and emphasize their difference. The color scales for node and link abundance (see Fig. 6.3(e)) are divided into four levels; the first level encodes zero abundance, the second level encodes a single node or link, and the remaining levels follow a log scale.

Highlighting Highlighting is enabled via mouse-over of up to two matrix rows, columns, or diagram contours. If two of such elements belong to the same group, then this group is highlighted in red (and translucent red in the background), as shown in Fig. 6.1. If two different groups are hovered, then one group is shown in red, the other group in blue, and their overlap in black (see Fig. 6.3). This simultaneous highlighting enables the comparison of two groups in coordinated views, which include additional information (task *A*).

One such coordinated view is shown at the right of Fig. 6.1, which lists the nodes of a highlighted node group, or those nodes covered by a highlighted link group. Three lists are shown when two groups are highlighted: two lists that show all nodes per group, and one list in the middle that shows overlapping nodes. Links are coupled to time series data in another view (see Fig. 6.7). It visualizes the time series of up to two highlighted groups as semi-transparent colored trend plots. For a highlighted link group, these are the time series that belong to its links, and for a highlighted node group, these are the time series of the links that cover any of its nodes. Time series of links that are shared by two groups are emphasized as black plots.

6.9. Exploration Demonstration

We demonstrate the described concepts and the use of the tool by exploring a dense network of countries (20 nodes) and their trade relations (95 links). These trade relations are the combined import and export (in millions of dollars) between countries, measured on a yearly basis between 1948 and 2000 (53 time points).

Preprocessing The analyzed data set was derived from a larger trade network by selecting the countries from four major regional trading groups (North-America as *AME*, Europe as *EUR*, Arabia as *ARA*, and Asia as *ASI*). In addition, trade relations were filtered for high variance, which leaves the most dynamic relations for analysis. These trade relations were attached to corresponding links as time series. Subsequently, links were grouped according to these time series with a k-means algorithm. This clustering uses angular (cosine) distance for the time series vectors such that links with similar trade dynamics are grouped together. Links from the same group could therefore be trade relations with common external influences, explaining their similar dynamics, or they could have been grouped as a clustering artifact. Clustering was applied twice to get a multi-level grouping, with $k = 5$ for top groups and $k = 3$ for subgroups. The resulting five top link groups can be typified as follows: a large group with consistent trade growth but a lot of missing data, a large group with consistent growth and little missing data, a smaller group with a minor trade slump in the 1980s, a group of four links with major slumps in the '90s, and a group of a single link with the same slump in the '90s.

Matrix perspective Dual adjacency matrix and node-link-contour visualizations of the initial, top-level link groups are shown in Fig. 6.1. These visualizations provide insights into the trade network topology. For example, the matrix at the top-left shows that all link groups overlap, which indicates that different trade behaviors intermingle. Three link groups at the top-left have a strong overlap (they share many nodes), which is shown by dark brown matrix intersections (T4).

The bottom-left and top-right matrix quadrant confirm a strong overlap of these link groups. These quadrants also reveal a large node group, led by the United Kingdom (*EUR_UKG*), that is exclusively covered by these link groups (T1). Only two other groups of countries, led by the United States (*AME_USA*) and Japan (*ASI_JPN*), are covered by more link groups, making them link group hubs (T6). The familiar adjacency matrix at the bottom-right shows that both country groups are also hubs in the traditional sense, connecting to most or all other node groups. However, China is isolated to one link group, making this link group a bridge (T7), while the bottom right matrix shows that China has links to many countries, making China a conventional hub that is constrained to one link group.

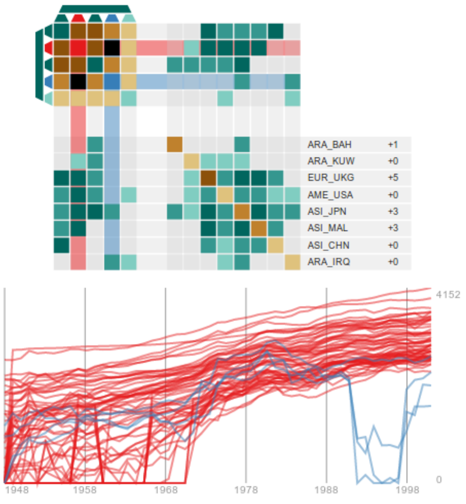
Node-link-contour perspective The node-link-contour diagram in Fig. 6.1 enables the same observations as the dual adjacency matrix because the link group topology is plain. However, the overlap between all link groups is not as apparent as in the top-left matrix. Likewise, the presence of three large, mostly overlapping, link groups is more striking in the bottom-left matrix than in the diagram. On the other hand, sparsely connected node groups such as Iraq (*ARA_IRQ*) and Bahrain (*ARA_BAH*) are easier to spot in the diagram than in the matrices. Moreover, small link groups stand out, such as the two groups that bridge to Iraq. These link groups can also be detected in the bottom-left matrix due to their sparse and aberrant covering of countries.

Inspecting link groups The three large link groups and two smaller link groups can be explained by inspecting their associated time series in an additional view (task *A*). In Fig. 6.7(a) we highlight one of the larger link groups (colored red) and one of the smaller groups (blue) by hovering their intersection in the top-left matrix quadrant. The large group contains a large number of links that appear as numerous red trend lines. These series are noisy until 1970, likely caused by a lack of data (defaulting trade volume to nil), but show consistent trade growth afterwards. Comparing the large link groups to each other confirms that all large link groups share this growth pattern. The smaller group shows a significant trade slump in the 90's. The bottom-left matrix quadrant shows that this link group covers Iraq.

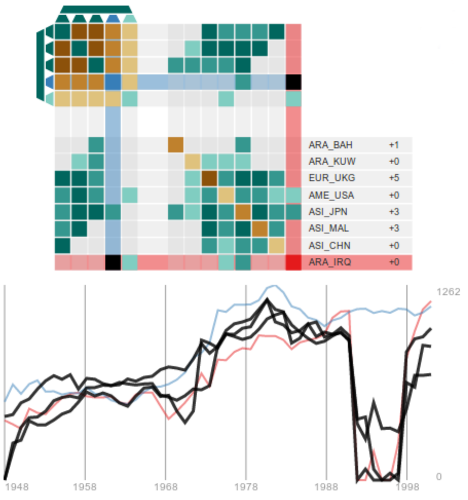
We request more detail about Iraq in Fig. 6.7(b) (colored red) by hovering its overlap with a link group (blue). Hovering the row or column of a group of countries includes all of its trade relations in the time series view. Trade volume plots of the blue link group that involve Iraq are colored solid black like the matrix overlap cell. We see that all of Iraq's relations are covered by the blue link group except for one (the red plot). Nonetheless, all of Iraq's trade relations feature the 90's volume slump, which is likely caused by the regional conflicts during this period.

Cluster assessment In Fig. 6.7(a)&(b) the blue link group contains one trade relation that does not slump and which excludes Iraq. This mismatch appears to be a fault in the clustering, so we split the blue link cluster by clicking on its link hierarchy branch at the top-left to get the new matrix configuration in Fig. 6.7(c). This unveils that the link without the slump, positioned between the blue and red subgroups, does not involve Iraq but Saudi Arabia (*ARA_SAU*) and France (*EUR_FRN*), which explains the trade development that is similar to Iraq up to the war and why their trade links were clustered together. Comparing one of the link subgroups that involve Iraq with the only other

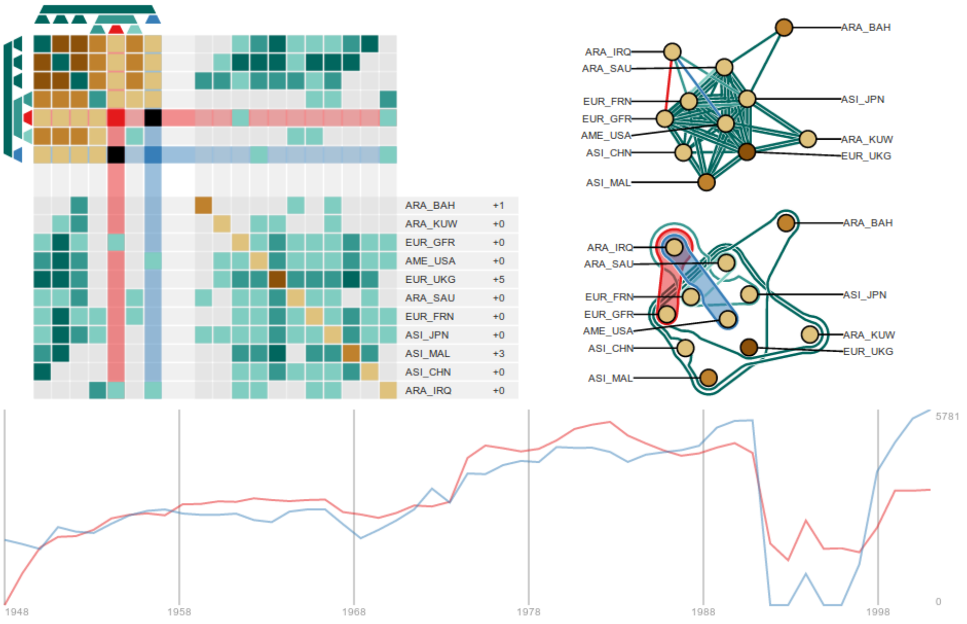
Figure 6.7: *Elucidation of link groups in a trade network, in which time series of highlighted groups are shown as trend plots that show trade volume on an adaptive log scale: (a) Comparison of the time series of a large link group (red) that has a mostly steady growth of trade volume, and a smaller link group (blue) with a significant trade slump in the 90's. (b) Overlap of all available trade relations of Iraq (red) with the relations of the smaller link group of (a) (blue), showing that many 90's volume slumps involve trade with Iraq (overlapping relations colored black). (c) The blue link group of (a) is expanded to reveal its three subgroups, of which one subgroup contains only one link (red, covering* ARA_IRQ *and* EUR_GFR*) that is compared to the single link of a smaller top group (blue, covering* ARA_IRQ *and* AME_USA*).*


ARA_BAH +1
ARA_KUW +0
EUR_UKG +5
AME_USA +0
ASI_JPN +3
ASI_MAL +3
ASI_CHN +0
ARA_IRQ +0
4152
0
1948
1958
1968
1978
1988
1998

(a)


ARA_BAH +1
ARA_KUW +0
EUR_UKG +5
AME_USA +0
ASI_JPN +3
ASI_MAL +3
ASI_CHN +0
ARA_IRQ +0
1262
0
1948
1958
1968
1978
1988
1998

(b)


ARA_BAH +1
ARA_KUW +0
EUR_GFR +0
AME_USA +0
EUR_UKG +5
ARA_SAU +0
EUR_FRN +0
ASI_JPN +0
ASI_MAL +3
ASI_CHN +0
ARA_IRQ +0
ARA_BAH
ARA_IRQ
ARA_SAU
ASI_JPN
EUR_FRN
EUR_GFR
AME_USA
ARA_KUW
ASI_CHN
EUR_UKG
ASI_MAL
5781
0
1948
1958
1968
1978
1988
1998

(c)

link group that involves Iraq (colored red and blue respectively in Fig. 6.7(c)) shows similar trade developments, but the isolation of the cluster (being the trade link with the United States) could be explained by its very strong increase in trade volume before the turn of the millennium.

6.10. Preliminary Expert Feedback

We performed interviews and walk-throughs of our prototype with two experts from different fields to gather feedback on our approach. We first performed a series of interviews with each expert to collect several of their datasets and associated analysis questions. Then, during a one-hour session, one of the authors walked the experts through the prototype while they used their own data.

Information science expert Our first expert, Jevin West, is an academic researcher at the information science school of the University of Washington, with solid expertise in analyzing complex networks and a focus on understanding communities. The datasets he was primarily interested in were scientific journal citation networks, composed of several hundreds of nodes and several thousands of links that are hierarchically partitioned into link communities [192]. After our presentation of the tool, it took about 15 minutes for Jevin to interact with the system and make effective use of the dual adjacency matrix to explore his data. Jevin commented that the tool had a steep learning curve, but is powerful once understood. This enables him to answer questions on link and node communities that were not easy to answer before. In particular, it helped him make the leap from link groups to shared node groups. For example, he identified nodes acting as hubs by glancing at the link to node group matrix and reviewing node groups. He also pointed out link communities sharing many nodes, commenting that this was helpful to identify noise in the data or regions with a low clustering quality.

Jevin was quickly familiar with the link group intersection matrix and used the hierarchical navigation several times to adjust the level of link groups to the desired granularity. Jevin concentrated most of his analysis on the link to node group matrix and the conventional node group matrix. He explained that “the most useful feature here [pointing at the node group adjacency matrix] is [seeing] what is not connected because we usually know about the [presence of] clusters [themselves]”. He commented that it was compelling to see holes in this matrix and argued that this could be useful to identify missing data or, if not missing, serve to make predictions on future connections, which he mentioned is of interest in many scenarios.

Neuroscience expert Our second expert, Tara Madhyastha, is an academic researcher at the radiology department of the University of Washington, with expertise in analyzing functional brain connectivity networks extracted from magnetic resonance imaging. She was primarily interested in dynamic weighted networks, composed of two dozens of regions of interest in the brain and their weighted connections that evolve over time. We augmented the prototype with a spatial brain view to provide a familiar context to the nodes and weighted connections. In our initial interview, Tara explained that she had experimented with several link grouping algorithms before. The output usually contained one large group of links and many small ones, from which she concluded it was not worth pursuing this type of analysis. Aware of these past attempts, we experimented with several other algorithms to get to a more balanced distribution of the number of links in groups. We presented the output of k-means clustering on vectors of link weight to Tara in our walk-through session. Tara had never before used visual tools to inspect link and node groups at the same time.

Tara had a very different exploration process than Jevin. She spent most of the session inspecting the content of the link group matrix coupled with the time series view. As we presented the tool, she immediately identified a link group of interest, pointing at the time series view, and asked a series of questions about the clustering algorithm. From this point, Tara used the prototype to visually assess the quality of the link groups. In a later session we combined Tara's own link clusters of three large-scale intrinsic brain zones [208] for her to explore a subset of individuals with Parkinson's Disease (PD) and age-matched controls. Research, including Tara's, suggests that the coupling of these zones may be dysregulated in PD. The DAM enabled her to examine the time-courses of multiple link communities in multiple zones at the same time, expecting that some are coupled and others are not (see Fig. 6.8). For example, she discovered partial coupling of the Posterior Default Mode zone and parts of the Fronto-Parietal Task Control zone in a PD subject at rest.

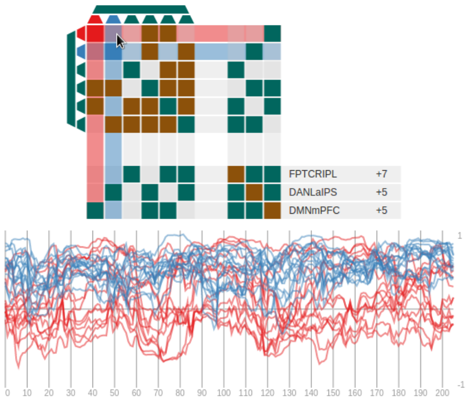
6.11. Discussion and Limitations

We have demonstrated DAMs for small but dense networks where links are subject to multi-level grouping. Even for networks with few nodes this poses a visualization challenge, because standard node-link diagrams are hard to read (see Fig. 6.7(c)). However, such a diagram is likely to enable more effective analyses for a sparse network, provided its link groups are coherent (see Fig. 6.6).

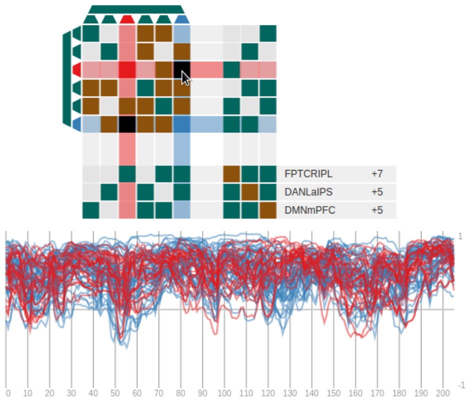
DAMs scale to larger networks by adding interaction techniques to manipulate link groups and their hierarchy: filtering link groups on criteria such as size and density, interactive branch pruning, and automated branch expansion. This involves common hierarchy interaction that is peripheral to the concept of DAMs but was vital to support Jevin's large link community analyses with our prototype. Nonetheless, analyses with DAMs can be impeded by flat or imbalanced link hierarchies.

An alternative to DAMs of dense networks are regular adjacency matrices with links color coded by group, and rows and columns arranged by link group similarity. While this regular adjacency matrix might be easier to read, color coding limits the number of link groups shown and complicates interactive link group navigation.

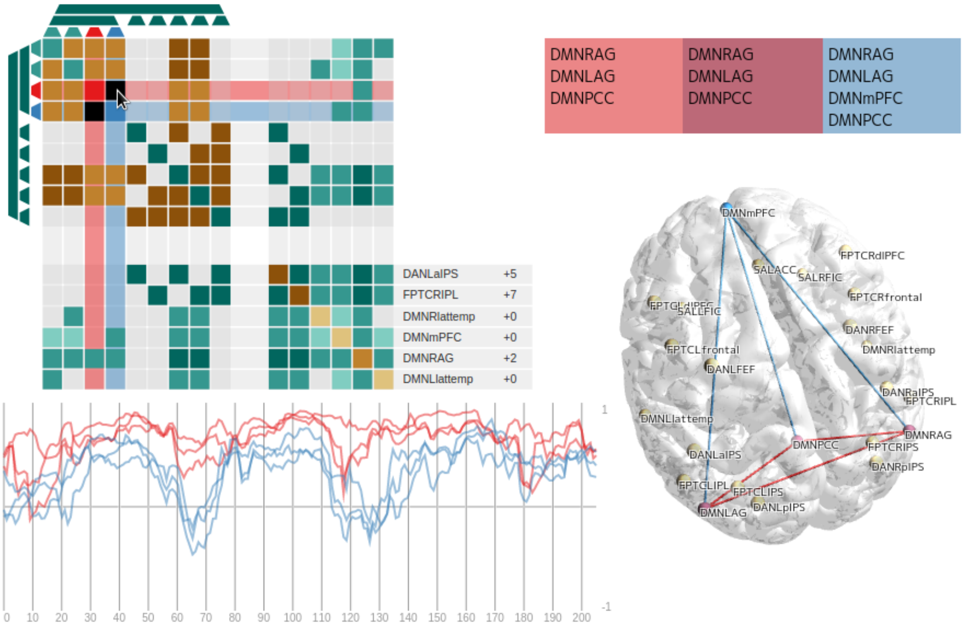
One benefit of a DAM is its dual representation of link groups, which enables the attachment of explicit hierarchy representations, and the mouse-over highlighting and comparison of two groups.



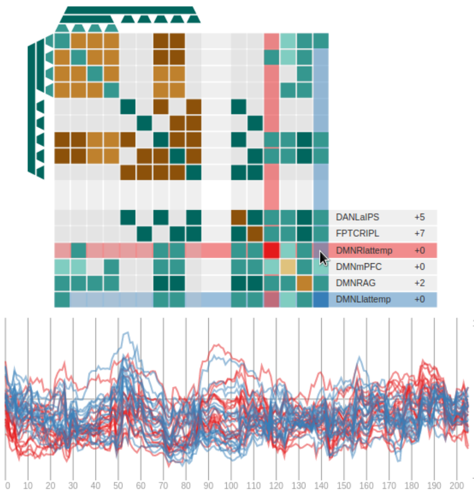
(a)



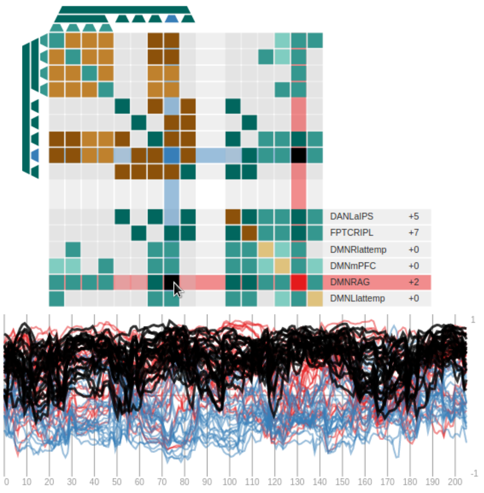
(b)



(c)



(d)



(e)

Figure 6.8: *Exploration of 20 brain regions connected by 183 links that encode (sliding window) fMRI signal correlations along 200 time points. The brain is divided into three zones (DAN, FPTC, and DMN) and links are grouped accordingly: three intra-zone link groups that are subgrouped [208], and three inter-zone link groups: (a) The top-left matrix shows no overlap between the three intra-zone link groups, but that they do overlap with the inter-zone link groups. The bottom-right matrix has three node groups that match the zones. The intra-zone link group DMN (red) has mixed signals, while DAN (blue) and FPTC have positive signals. (b) Both the FPTC link group (red) and the link group between FPTC and DAN (blue) have positive signals. (c) Splitting DMN because of its mixed signals reveals its overlapping link subgroups. Two link subgroups have strong overlap, where one group (red) is strongly correlated and tightly positioned in the brain, and the other group (blue) is less correlated and more spread across the brain. (d) The bottom-right adjacency matrix shows a missing link between spatial opposites Rlattemp and Llattemp. Hovering the empty spot compares all neighboring links of Rlattemp and Llattemp, where their signals show a consensus. (e) One node group (red) acts as a hub to DMN link groups. Hovering this node group and the DMN link groups shows that the node group has many anti-correlations within the DMN zone. However, hovering the inter-zone link groups (blue) shows that this node group has mostly positive correlations with the 'remainder' of the network.*

6.12. Conclusion

We have introduced a generalization of the adjacency matrix for exploring link and node groups. This generalization enables analysis of (hierarchical) link groups while providing both node and link group perspectives. Iterative implementation of this concept, while relying on feedback from link group experts, has resulted in an interactive system with coordinated matrix and node-link diagram views. Walk-through sessions with two experts revealed that DAMs help link and node group analysis, bridging the concepts.

The experts had different exploration processes, in which they understood and relied on all quadrants of the adjacency matrix in spite of a steep learning curve. This feedback suggests that our approach enables analysts to bridge the gap between link and node groups. We believe this is an encouraging first step towards visual exploration of link groups.

Acknowledgements We thank Jevin West and Tara Madhyastha for their expert feedback. We also thank Sune Lehmann and Yong-Yeol Ahn for their constructive comments about earlier prototypes. This research was supported by grants from the National Institutes of Health (NIH) 1RC4NS073008-01 and P50NS062684K, and Netherlands Organization for Scientific Research (NWO) 612.001.004.

7
Retrospective

The techniques presented in the previous chapters concern modular network structures in various forms. We sometimes talk about these modular structures as subnetworks. Subnetworks are generic, e.g., the interactive highlighting of a node and its neighbors *is* the portrayal of a subnetwork within a subnetwork. What differentiates our techniques are a focus on particular modular structures in biology and leveraging associated modular structures to aid biologists in their analyses. Moreover, the simultaneous portrayal of multiple network structures recurs in all chapters:

Compressed Adjacency Matrices were introduced in Chapter 3 to make gene regulatory motifs (frequently occurring network structures) stand out in context of the entire network. These motifs are of interest to biologists because of their concerted control of cell processes.

Kelp Diagrams can be used to portray multiple set relations as schematic contours on top of existing visualizations, as was shown for a metabolic network at the beginning of Chapter 4.

eXamine is a tool that enables the interactive exploration of many ranked annotations on top of a small protein interaction module. The methodology presented in Chapter 5 targets modules that are small and sparse due to their extraction from larger networks via protein expression analysis.

Dual Adjacency Matrices enable the analysis of multiple structures that are induced via an unconventional perspective; the grouping of links instead of nodes. In Chapter 6 we have shown how link groups are of benefit to the analysis of dense networks, such as fMRI brain networks.

The above techniques share a theme but diverge in assumptions about the kinds of structures present in the data that is to be analyzed, and the analysis goals themselves. The proceeding chapters describe each technique in isolation, often skipping details of the underlying research process, pitfalls, and surprises. These details are left out to maintain clarity and focus. Luckily, it is not too late to reminisce. I therefore discuss some of these details, placed in context of larger issues that I have encountered during my research.

Addressing Challenges

In Chapter 2 we identified several challenges for the visualization of gene regulation, protein interaction, and metabolic networks. Some of these challenges are addressed by the introduced techniques. For example, we can now convey local GRN motifs and neighborhood patterns in an integrated and structured overview with CAMs, though supporting the following of paths remains a challenge. We proposed a similar tactic for PINs in Chapter 2, which would have been close to the NodeTrix approach due to PINs' undirected edges. Instead of testing this tactic, we aimed at the more urgent challenge of visualizing smaller PIN modules via eXamine, prioritizing the depiction of detailed PIN topology and annotations over motif and neighborhood patterns in larger PINs.

We have not addressed most of the metabolic network challenges. Yet, metabolic pathways do feature as PIN annotations in Chapter 5. Interestingly, the eXamine case study suggests that the metabolic network analysis challenges should be shifted in priority from pathway-centric (for example, the visual integration of multiple pathways) to a more heterogeneous data landscape that puts GRNs, PINs, and metabolic networks on an even footing. This shift was already reflected in the challenge of network integration, which reappears in eXamine where we show that supporting the integration of networks is not always about combining large networks, but about the navigability of the many inter-network associations of smaller modules.

We were unable to come to grips with several challenges of Chapter 2 (network alignment in particular) because these appear in an generic and isolated form. For example, solving 'the PIN visualization' challenge is difficult without placing constraints on analysis questions that are to be answered and whether additional PIN information is available that can help in finding answers. eXamine addresses a specific form of a PIN visualization challenge, not PIN visualization in its entirety. More time should therefore be invested in cataloging the specifics of biological network analysis, as opposed to creating taxonomies of biological network fundamentals, which are easier to interpret by computer scientists, but less meaningful for biologists.

Everything and Nothing is a Network

Computer scientists often strive for the solution to a generic problem, even when the instigating problem is more specific. However, taking a generic viewpoint and abstracting from the specifics of a problem could result in missed opportunities due to a lack of context. The expression "Everything is a network" is often uttered by network scientists, but lies within a stone's throw from "Everything is an object". For example, it is possible to combine any two objects into an abstract group of objects, but we may want to take precautions if we know these two objects to be a jerry can and a lighter.

Targeting data characteristics and associated analysis questions lies at the heart of visualization. However, dealing with (domain) specifics may have been lost with the rise (or hype) of generic network analysis. Being able to translate observations into networks for further analysis can be of great use. Yet, we should also be concerned about the information loss, or distortion in reasoning, that results from this translation.

Our way of dealing with gene regulatory networks in Chapter 3 demonstrates that targeting networks with specific characteristics can benefit visualization design and provide new perspectives. However, in doing so we have still ignored the source of GRNs: the positions of genes on a chromosome and the physical interactions of gene products. These two aspects play an important role in (indirect) regulations between genes that are encoded by the links of a GRN.

Case in point is the scale-free outbound degree distribution of nodes in a GRN. This is a nice characteristic for mathematicians to discover, and for us to exploit in the compression of adjacency matrices. However, this distribution is strongly related to how genes are positioned on a chromosome and how their synthesis is inhibited or promoted by gene products that latch onto the chromosome at a nearby position. Genes that are positioned closely together on a chromosome are therefore likely to be under the influence of the same genes and their products. It might be possible to derive a better composition of the compressed adjacency matrix (which includes the stacking of nodes) directly from the positions of genes on the chromosome. Finally, it leaves us with an additional question: To what extent are we reconstructing the chromosomal positions of genes from GRNs by computing the CAM arrangement?

These observations do not contradict the importance of abstract reasoning and generic solutions. For example, we initially observed the peculiar structural characteristics of gene regulatory networks by looking at their node-link diagrams, which were created with a generic visualization technique. It was only later on that we confirmed these characteristics via targeted computations and existing literature. Moreover, in Chapter 5 we rely on a generic combinatorial optimization technique to extract a protein module.

Staking Stakeholders

Visualization techniques can not only be generic in the data that they are able to convert into images, but also in the analysis tasks that they support. In Chapter 3 we focus on the ability of the analysts to find GRN motifs, which is quite specific from the perspective of a computer scientist. However, providing an overview of these structures is likely of greater interest to a bioinformatician than to a biologist. In Chapter 5 we observe that biologists are far more interested in network specifics, where the legible depiction of important proteins and their interactions outweigh the provision of context by displaying additional information.

Both node-link diagrams (Kelp and eXamine) and adjacency matrices (CAMs and

DAMs) have been used to present networks throughout this thesis. Predictably, we use node-link diagrams for network and set system topologies that we expect to be simple, and we use adjacency matrices for complex topologies. The users targeted by each encoding (and underlying topology complexity) divide along this line as well. CAMs of GRNs are of most interest to bioinformaticians, who themselves develop analysis techniques. Likewise, DAMs were originally developed around link community detection algorithms (before transitioning to a more generic form) and the designers of these algorithms were the ones who appreciated DAMs the most. On the other hand, Kelp Diagrams and eXamine target observers without deep knowledge of any underlying data analysis algorithms; the knowledge that they can derive from the data is their goal, instead of knowledge about preprocessing algorithms.

The generic set system analysis tasks that are introduced in Chapter 4 serve as an evaluation tool for Kelp Diagrams and its competitors. These tasks are modeled directly on set systems and are therefore highly abstract, ignoring possible stakeholders and their specific analysis needs. A senior Dutch visualization researcher brought this to my attention when I presented Kelp Diagrams at a conference, pointing out that an attempt to service everyone by focusing on generic analysis goals could spread a solution too thin. This is a possible outcome, but also suggests that a visualization technique should have an immediate purpose. Instead, the analysis tasks and methodology of Kelp served as a scaffold for dealing with set systems in eXamine and Dual Adjacency Matrices. Moreover, the case study of Chapter 5 shows that Kelp's abstract tasks have relevant, concrete counterparts.

Graph Visualization and Network Drawing

There are two scientific domains that deal with the visualization of networks: *Visualization* and *Graph Drawing*. The visualization domain is mostly concerned with interactive visual analysis of any data type, networks included, and its research covers a broad spectrum between the abstract and concrete. This has resulted in a rich collection of techniques that target specific and combined data forms, which is of particular use for the kind of research questions that biologists pose in light of available omics data. The graph drawing domain is concerned with efficient and effective drawing of networks (or graphs), often achieved by imposing aesthetic optimization goals on the embedding of node-link diagrams. Kelp Diagrams were developed in the same manner.

These two domains are similar but differ in their priorities. Scalability and flexibility is important in the visualization domain, which stimulates the use of simple encodings and layout algorithms. Graph drawing, on the other hand, prioritizes (theoretical) optimization of aesthetic constraints and criteria. This stimulates the development of complicated algorithms that often take only small networks as input. These priorities (arguably) stem from the domains' origins.

This difference in priorities seems to make the domains incompatible. Yet, certain scenarios reveal the complementary nature of graph drawing and visualization. Case in point is the analysis of large Protein Interaction Networks with eXamine in Chapter 5. The discussed PIN that consists of around 4000 proteins, the differential expression of corresponding genes, and 30,000 interactions, could be visualized in Cytoscape as a node link diagram that is laid out with a plain force simulation. The resulting visualization is impressive at first and provides a fair depiction of the data complexity. These visualizations therefore often serve as billboards, on websites and even papers. Supporting specific analyses is more involved however.

Detecting a small, differentially expressed module in a large, highly connected network, is difficult to do by eye alone and prone to human error. eXamine therefore uses automated and concise module extraction, discarding most of the network as irrelevant with respect to experimental conditions, which leaves a small and sparse network of proteins. The small size and complexity of such a module makes the generation of a high-quality diagram with a graph drawing algorithm viable. Moreover, as we have seen in Chapter 5, the analysis of these modules relies heavily on the observation of specific proteins and interactions, as opposed to the observation of clusters and outliers in network topology. The use of automated analysis therefore mitigates the PIN scalability issue, but increases the need for a tidier depiction of what remains. The layout algorithm that we developed for eXamine is a means to this end; it is simple but able to combine multiple network structures (a PIN topology and a set system of annotations). Nonetheless, a more refined algorithm can create tidier node-link diagrams and therefore enable more efficient analyses.

Numerous network visualization techniques incorporate filtering and aggregation techniques to maintain legibility or introduce conciseness as data volume scales. Graph drawing can therefore benefit network visualization by providing the means to make tidy drawings when the data characteristics and the concerns of analysts merit this. (Proving theoretical characteristics of drawings and algorithm running times is a different matter, however.) On the other hand, visualization provides graph drawing with scenarios that make it of practical use in interactive data exploration.

Drifting Analysis Goals

We can throw ourselves at specific analysis tasks in an attempt to make visualizations as effective as possible. However, building a visualization tool to suit the goals of a user, who subsequently uses the tool, may affect the user and shift the analysis goals considerably.

For example, the initial prototypes of eXamine were not designed to provide a semantic context for extracted protein modules. Instead, we aimed to investigate the effects of the False Discovery Rate parameter on the size and other characteristics of the extracted module, and how many modules were extracted. The extracted modules, per FDR step within a predefined range, formed a set system on top of the union of all modules. We expected a consistency between modules for consecutive FDR values, and therefore designed the SOM layout algorithm, which would flatten out high-dimensional redundancies and provide set contours as a bonus.

The visualization technique itself worked but showed that the module extraction algorithm was more predictable than we had anticipated; one or two proteins were added to the module per FDR increment. Such stable behavior is, of course, desired in case of module detection, and indicates that the data contains a clear signal. However, it does not make for an interesting demonstration of a visualization technique. Luckily, our curiosity as computer scientists consequently shifted to the explanation of this strong signal, which we were able to do by integrating domain knowledge via enrichment analysis.

The simultaneous visualization of a network and a set system therefore, in the end, serve a purpose that is different from its initial design goals (and motivates the development of generic techniques). It was only at this point that we exposed our biologist collaborators to a prototype of eXamine, which resulted in the findings of Chapter 5. These findings make for the most convincing visualization case in this thesis, which is remarkable because the inception of eXamine was mostly technique-driven, where specific analysis questions materialized late in the development process. This shows that visualization as a practice is not only useful by providing analysis tools, but also by stimulating the research process itself and the people involved.

Directing the Drift

In light of drifting analysis goals I still wonder how visualization as a discipline can best service cell biology research. We should of course continue to solve specific research problems via visualization. Even if research goals shift in the end, we will have gained understanding and we will have contributed to the research process. However, based on the aforementioned experiences, my intuition directs me into a research direction that could be of greater value to cell biologists.

We must empower the analysis methodology that has proven so effective for eXamine, namely faceted exploration and inclusion of network annotations, which taps into the uniquely rich stores of knowledge that are at the disposal of cell biology researchers. In order to do so, we will have to retain the familiar node-link-contour encoding that has been the bread and butter of cell biology illustrations for decades (see Chapter 2). However, at the same time we will have to integrate (domain-specific) annotation navigation more tightly into a network view, instead of a side view.

More specifically, we have to improve scalability and aesthetics by replacing the SOM layout algorithm of Chapter 5 with an exact combinatorial optimization method. Here we will also regard binary edges as a special case of sets, and consequently links as a tightened form of contour, in order to specify a uniform optimization criterion.

In addition, we should use the ability to visualize both network and set systems to compress and mark modular structures losslessly with the Power Graph method (see Fig. 2.9). Here we have to experiment with different ways to conform link and node compression to already known or tagged annotations such that this compression is no longer ambiguous and easier to interpret.

I hope to have convinced you that the discussed research is but a stepping stone in the support of complex analyses. I therefore delegate my closing statement to a wiser man who, without attaining a formal degree, became the leading consulting engineer of the United States during its antebellum era:

> No engineer can go upon a new work and not find something peculiar, that will demand his careful reflection, and the deliberate consideration of any advice that he may receive; and nothing so fully reveals his incapacity as a pretentious assumption of knowledge, claiming to understand everything.
>
> - John B. Jervis (1795-1885)

References

[1] G. Michal and D. Schomburg, *Biochemical pathways: an atlas of biochemistry and molecular biology*. Wiley, 1999.

[2] P. Shannon, A. Markiel, O. Ozier, N. Baliga, J. Wang, D. Ramage, N. Amin, B. Schwikowski, and T. Ideker, "Cytoscape: A software environment for integrated models of biomolecular interaction networks," *Genome Research*, vol. 13, no. 11, pp. 2498-2504, 2003.

[3] M. Suderman and M. Hallett, "Tools for visually exploring biological networks," *Bioinformatics*, vol. 23, no. 20, pp. 2651-2659, 2007.

[4] G. Pavlopoulos, A. Wegener, and R. Schneider, "A survey of visualization tools for biological network analysis," *BioData Mining*, vol. 1, no. 1, p. 12, 2008.

[5] I. Herman, G. Melançon, and M. Marshall, "Graph visualization and navigation in information visualization: A survey," *IEEE Trans. on Visualization and Computer Graphics*, vol. 6, no. 1, pp. 24-43, 2000.

[6] T. von Landesberger, A. Kuijper, T. Schreck, J. Kohlhammer, J. van Wijk, J.-D. Fekete, and D. Fellner, "Visual analysis of large graphs: State-of-the-art and future research challenges," *Computer Graphics Forum*, vol. 30, no. 6, pp. 1719-1749, 2011.

[7] M. Graham and J. Kennedy, "A survey of multiple tree visualisation," *Information Visualization*, vol. 9, no. 4, pp. 235-252, 2010.

[8] R. Amar, J. Eagan, and J. Stasko, "Low-level components of analytic activity in information visualization," *IEEE Symp. on Information Visualization*, pp. 111-117, 2005.

[9] B. Lee, C. Plaisant, C. Parr, J.-D. Fekete, and N. Henry, "Task taxonomy for graph visualization," *Proc. of the AVI Workshop on Beyond Time and Errors: Novel Evaluation Methods for Information Visualization*, pp. 1-5, 2006.

[10] M. Smooth, K. Ono, J. Ruscheinski, P. Wang, and T. Ideker, "Cytoscape 2.8: new features for data integration and network visualization," *Bioinformatics*, vol. 27, no. 3, pp. 431-432, 2011.

[11] Z. Hu, D. Ng, T. Yamada, C. Chen, S. Kawashima, J. Mellor, B. Linghu, M. Kanehisa, J. Stuart, and C. DeLisi, "VisANT 3.0: New modules for pathway visualization, editing, prediction and construction," *Nucleic Acids Research*, vol. 35, no. suppl 2, pp. W625-W632, 2007.

[12] M. Baitaluk, M. Sedova, A. Ray, and A. Gupta, "BiologicalNetworks: Visualization and analysis tool for systems biology," *Nucleic Acids Research*, vol. 34, pp. W466-W471, 2006.

[13] S. Kozhenkov, Y. Dubinina, M. Sedova, A. Gupta, J. Ponomarenko, and M. Baitaluk, "BiologicalNetworks 2.0 - an integrative view of genome biology data," *BMC Bioinformatics*, vol. 11, no. 1, p. 610, 2010.

[14] B. Junker, C. Klukas, and F. Schreiber, "VANTED: A system for advanced data analysis and visualization in the context of biological networks," *BMC Bioinformatics*, vol. 7, no. 1, p. 109, 2006.

[15] M. Westenberg, S. van Hijum, O. Kuipers, and J. Roerdink, "Visualizing genome expression and regulatory network dynamics in genomic and metabolic context," *Computer Graphics Forum*, vol. 27, no. 3, pp. 887-894, 2008.

[16] G. Battista, P. Eades, R. Tamassia, and I. Tollis, *Graph drawing: algorithms for the visualization of graphs*. Prentice Hall PTR, 1998.

[17] J. Díaz, J. Petit, and M. Serna, "A survey of graph layout problems," *ACM Computing Surveys*, vol. 34, no. 3, pp. 313-356, 2002.

[18] P. Eades, "A heuristic for graph drawing," *Congressus numerantium*, vol. 42, pp. 149-160, 1984.

[19] T. Fruchterman and E. Reingold, "Graph drawing by force-directed placement," *Software: Practice and Experience*, vol. 21, no. 11, pp. 1129-1164, 1991.

[20] D. Archambault, T. Munzner, and D. Auber, "Topolayout: Multilevel graph layout by topological features," *IEEE Trans. on Visualization and Computer Graphics*, vol. 13, no. 2, pp. 305-317, 2007.

[21] G. Klau and P. Mutzel, "Optimal compaction of orthogonal grid drawings," *Integer Programming and Combinatorial Optimization*, vol. 1610, pp. 304-319, 1999.

[22] D. Holten, "Hierarchical edge bundles: Visualization of adjacency relations in hierarchical data," *IEEE Trans. on Visualization and Computer Graphics*, vol. 12, no. 5, pp. 741-748, 2006.

[23] W. McCormick, P. Schweitzer, and T. White, “Problem decomposition and data reorganization by a clustering technique,” *Operations Research*, vol. 20, no. 5, pp. 993-1009, 1972.

[24] P. Saraiya, C. North, and K. Duca, “An evaluation of microarray visualization tools for biological insight,” *IEEE Symp. on Information Visualization*, pp. 1-8, 2004.

[25] G. Kerr, H. Ruskin, M. Crane, and P. Doolan, “Techniques for clustering gene expression data,” *Computers in Biology and Medicine*, vol. 38, no. 3, pp. 283-293, 2008.

[26] J. S. Yi, Y. ah Kang, J. Stasko, and J. Jacko, “Toward a deeper understanding of the role of interaction in information visualization,” *IEEE Trans. on Visualization and Computer Graphics*, vol. 13, pp. 1224-1231, 2007.

[27] R. Rao and S. Card, “The table lens: merging graphical and symbolic representations in an interactive focus+ context visualization for tabular information,” *SIGCHI Conf. on Human factors in computing systems*, pp. 318-322, 1994.

[28] A. Barabási and Z. Oltvai, “Network biology: understanding the cell’s functional organization,” *Nature Reviews Genetics*, vol. 5, no. 2, pp. 101-113, 2004.

[29] R. Albert, H. Jeong, and A. Barabási, “Error and attack tolerance of complex networks,” *Nature*, vol. 406, no. 6794, pp. 378-382, 2000.

[30] R. Cohen, S. Havlin, and D. ben Avraham, *Handbook of Graphs and Networks: From the Genome to the Internet*, ch. 4, pp. 85-110. Wiley, 2005.

[31] D. Rodionov, “Comparative genomic reconstruction of transcriptional regulatory networks in bacteria,” *Chemical reviews*, vol. 107, no. 8, pp. 3467-3497, 2007.

[32] T. Schlitt and A. Brazma, “Current approaches to gene regulatory network modelling,” *BMC bioinformatics*, vol. 8, no. Suppl 6, p. S9, 2007.

[33] L. Jensen, J. Saric, and P. Bork, “Literature mining for the biologist: from information retrieval to biological discovery,” *Nature reviews genetics*, vol. 7, no. 2, pp. 119-129, 2006.

[34] U. Alon, *An introduction to systems biology: design principles of biological circuits*. CRC press, 2007.

[35] S. Shen-Orr, R. Milo, S. Mangan, and U. Alon, “Network motifs in the transcriptional regulation network of escherichia coli,” *Nature genetics*, vol. 31, no. 1, pp. 64-68, 2002.

[36] R. Janky, J. Helden, and M. Babu, “Investigating transcriptional regulation: From analysis of complex networks to discovery of cis-regulatory elements,” *Methods*, vol. 48, no. 3, pp. 277-286, 2009.

[37] S. Bornholdt, “Boolean network models of cellular regulation: prospects and limitations,” *Journal of the Royal Society Interface*, vol. 5, no. Suppl 1, pp. S85-S94, 2008.

[38] K. Sugiyama, S. Tagawa, and M. Toda, “Methods for visual understanding of hierarchical system structures,” *IEEE Trans. on Systems, Man and Cybernetics*, vol. 11, no. 2, pp. 109-125, 1981.

[39] G. Pavlopoulos, S. O’Donoghue, V. Satagopam, T. Soldatos, E. Pafilis, and R. Schneider, “Arena3d: visualization of biological networks in 3d,” *BMC systems biology*, vol. 2, no. 1, p. 104, 2008.

[40] W. Longabaugh, “Biotapestry: A tool to visualize the dynamic properties of gene regulatory networks.,” *Methods in molecular biology*, vol. 786, no. 4, pp. 359-394, 2012.

[41] F. Schreiber, T. Dwyer, K. Marriott, and M. Wybrow, “A generic algorithm for layout of biological networks,” *BMC bioinformatics*, vol. 10, no. 1, p. 375, 2009.

[42] A. Bezerianos, P. Dragicevic, J.-D. Fekete, J. Bae, and B. Watson, “GeneaQuilts: A system for exploring large genealogies,” *IEEE Trans. on Visualization and Computer Graphics*, vol. 16, no. 6, pp. 1073-1081, 2010.

[43] Juhee Bae and B. Watson, “Developing and evaluating quilts for the depiction of large layered graphs,” *IEEE Trans. on Visualization and Computer Graphics*, vol. 17, no. 12, pp. 2268-2275, 2011.

[44] N. Henry, A. Bezerianos, and J.-D. Fekete, “Improving the readability of clustered social networks using node duplication,” *IEEE Trans. on Visualization and Computer Graphics*, vol. 14, no. 6, pp. 1317-1324, 2008.

[45] A. Lambert, J. Dubois, and R. Bourqui, “Pathway preserving representation of metabolic networks,” *Computer Graphics Forum*, vol. 30, no. 3, pp. 1021-1030, 2011.

[46] S. Fields and O. Song, “A novel genetic system to detect protein-protein interactions,” *Nature*, vol. 340, no. 6230, pp. 245-246, 1989.

[47] J. Rual, K. Venkatesan, T. Hao, T. Hirozane-Kishikawa, A. Dricot, N. Li, G. Berriz, F. Gibbons, M. Dreze, N. Ayivi-Guedehoussou, *et al.*, "Towards a proteome-scale map of the human protein-protein interaction network," *Nature*, vol. 437, no. 7062, pp. 1173-1178, 2005.

[48] L. Kiemer and G. Cesareni, "Comparative interactomics: comparing apples and pears?," *TRENDS in Biotechnology*, vol. 25, no. 10, pp. 448-454, 2007.

[49] H. Jeong, S. Mason, A. Barabasi, and Z. Oltvai, "Lethality and centrality in protein networks," *Nature*, vol. 411, no. 6833, pp. 41-42, 2001.

[50] A. Wagner, "The yeast protein interaction network evolves rapidly and contains few redundant duplicate genes," *Molecular Biology and Evolution*, vol. 18, no. 7, pp. 1283-1292, 2001.

[51] S. Yook, Z. Oltvai, and A. Barabási, "Functional and topological characterization of protein interaction networks," *Proteomics*, vol. 4, no. 4, pp. 928-942, 2004.

[52] L. Royer, M. Reimann, B. Andreopoulos, and M. Schroeder, "Unraveling protein networks with power graph analysis," *PLoS Computational Biology*, vol. 4, no. 7, p. e1000108, 2008.

[53] E. Zotenko, J. Mestre, D. O'Leary, and T. Przytycka, "Why do hubs in the yeast protein interaction network tend to be essential: reexamining the connection between the network topology and essentiality," *PLoS computational biology*, vol. 4, no. 8, p. e1000140, 2008.

[54] F. Luo, B. Li, X. Wan, and R. Scheuermann, "Core and periphery structures in protein interaction networks," *BMC bioinformatics*, vol. 10, no. Suppl 4, p. S8, 2009.

[55] D. Merico, D. Gfeller, G. Bader, *et al.*, "How to visually interpret biological data using networks," *Nature biotechnology*, vol. 27, no. 10, pp. 921-924, 2009.

[56] G. Wagner, M. Pavlicev, and J. Cheverud, "The road to modularity," *Nature Reviews Genetics*, vol. 8, no. 12, pp. 921-931, 2007.

[57] Z. Wang and J. Zhang, "In search of the biological significance of modular structures in protein networks," *PLoS Computational Biology*, vol. 3, no. 6, p. e107, 2007.

[58] O. Garcia, C. Saveanu, M. Cline, M. Fromont-Racine, A. Jacquier, B. Schwikowski, and T. Aittokallio, "Golorize: a cytoscape plug-in for network visualization with gene ontology-based layout and coloring," *Bioinformatics*, vol. 23, no. 3, pp. 394-396, 2007.

[59] A. Barsky, T. Munzner, J. Gardy, and R. Kincaid, “Cerebral: Visualizing multiple experimental conditions on a graph with biological context,” *IEEE Trans. on Visualization and Computer Graphics*, vol. 14, no. 6, pp. 1253-1260, 2008.

[60] N. Henry, J.-D. Fekete, and M. McGuffin, “NodeTrix: a hybrid visualization of social networks,” *IEEE Trans. on Visualization and Computer Graphics*, vol. 13, no. 6, pp. 1302-1309, 2007.

[61] G. Bader and C. Hogue, “An automated method for finding molecular complexes in large protein interaction networks,” *BMC Bioinformatics*, vol. 4, no. 1, p. 2, 2003.

[62] N. Henry and J.-D. Fekete, “MatrixExplorer: a dual-representation system to explore social networks,” *IEEE Trans. on Visualization and Computer Graphics*, vol. 12, no. 5, pp. 677-684, 2006.

[63] H. Jeong, B. Tombor, R. Albert, Z. Oltvai, and A. Barabási, “The large-scale organization of metabolic networks,” *Nature*, vol. 407, no. 6804, pp. 651-654, 2000.

[64] E. Ravasz, A. Somera, D. Mongru, Z. Oltvai, and A. Barabási, “Hierarchical organization of modularity in metabolic networks,” *Science*, vol. 297, no. 5586, pp. 1551-1555, 2002.

[65] P. Saraiya, C. North, and K. Duca, “Visualizing biological pathways: Requirements analysis, systems evaluation and research agenda,” *Information Visualization*, vol. 4, no. 3, pp. 191-205, 2005.

[66] V. Lacroix, L. Cottret, P. Thébault, and M. Sagot, “An introduction to metabolic networks and their structural analysis,” *IEEE/ACM Trans. on Computational Biology and Bioinformatics*, vol. 5, no. 4, pp. 594-617, 2008.

[67] R. Caspi, T. Altman, J. Dale, K. Dreher, C. Fulcher, F. Gilham, P. Kaipa, A. Karthikeyan, A. Kothari, M. Krummenacker, M. Latendresse, L. Mueller, S. Paley, L. Popescu, A. Pujar, A. Shearer, P. Zhang, and P. Karp, “The MetaCyc database of metabolic pathways and enzymes and the BioCyc collection of pathway/genome databases,” *Nucleic Acids Research*, vol. 38, no. suppl 1, pp. D473-D479, 2010.

[68] H. Ogata, S. Goto, K. Sato, W. Fujibuchi, H. Bono, and M. Kanehisa, “KEGG: Kyoto encyclopedia of genes and genomes,” *Nucleic Acids Research*, vol. 27, no. 1, pp. 29-34, 1999.

[69] K. Kojima, M. Nagasaki, and S. Miyano, “An efficient biological pathway layout algorithm combining grid-layout and spring embedder for complicated cellular location information,” *BMC Bioinformatics*, vol. 11, no. 1, p. 335, 2010.

[70] M. Becker and I. Rojas, “A graph layout algorithm for drawing metabolic pathways,” *Bioinformatics*, vol. 17, no. 5, pp. 461-467, 2001.

[71] F. Schreiber, “High quality visualization of biochemical pathways in BioPath,” *In Silico Biology*, vol. 2, no. 59-73, 2002.

[72] P. Karp and S. Paley, “Automated drawing of metabolic pathways,” *3rd Int. Conf. Bioinformatics and Genome Research*, pp. 225-238, 1994.

[73] S. Hachul and M. Jünger, “Drawing large graphs with a potential-field-based multilevel algorithm,” in *Graph Drawing*, vol. 3383 of *Lecture Notes in Computer Science*, pp. 285-295, 2005.

[74] P. Karp, M. Riley, M. Saier, and I. Paulsen, “The ecocyc and metacyc databases,” *Nucleic Acids Research*, vol. 28, no. 1, pp. 56-59, 2000.

[75] D. Auber, “Tulip : A huge graph visualisation framework,” *Graph Drawing Software, Mathematics and Visualization*, pp. 105-126, 2004.

[76] R. Bourqui, V. Lacroix, L. Cottret, D. Auber, P. Mary, M.-F. Sagot, and F. Jourdan, “Metabolic network visualization eliminating node redundance and preserving metabolic pathways,” *BMC Systems Biology*, vol. 1, no. 1, p. 29, 2007.

[77] C. Klukas and F. Schreiber, “Dynamic exploration and editing of KEGG pathway diagrams,” *Bioinformatics*, vol. 23, no. 3, pp. 344-350, 2007.

[78] 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 *Graph Drawing*, vol. 5849 of *Lecture Notes in Computer Science*, pp. 256-267, 2010.

[79] J. Branke, “Dynamic graph drawing,” in *Drawing Graphs*, vol. 2025 of *Lecture Notes in Computer Science*, pp. 228-246, 2001.

[80] H. Kitano, *Foundations of Systems Biology*. MIT Press, 2001.

[81] H. Kitano, “Computational systems biology,” *Nature*, vol. 420, no. 6912, pp. 206-210, 2002.

[82] E. Klipp, R. Herwig, A. Kowald, C. Wierling, and H. Lehrach, *Systems Biology in Practice. Concepts, Implementation and Application*. John Wiley & Sons, 2008.

[83] R. Jianu, K. Yu, L. Cao, V. Nguyen, A. Salomon, and D. Laidlaw, “Visual integration of quantitative proteomic data, pathways, and protein interactions,” *IEEE Trans. on Visualization and Computer Graphics*, vol. 16, no. 4, pp. 609-620, 2010.

[84] J. Swedlow, S. Lewis, and I. Goldberg, “Modelling data across labs, genomes, space and time,” *Nature Cell Biology*, vol. 8, no. 11, pp. 1190-1194, 2006.

[85] M. Baitaluk, S. Kozhenkov, Y. Dubinina, and J. Ponomarkeno, “IntegromeDB: an integrated system and biological search engine,” *BMC Genomics*, vol. 13, no. 1, p. 35, 2012.

[86] C.-H. Yeang and M. Vingron, “A joint model of regulatory and metabolic networks,” *BMC Bioinformatics*, vol. 7, no. 1, p. 332, 2006.

[87] N. Tenazinha and S. Vinga, “A survey on methods for modeling and analyzing integrated biological networks,” *IEEE/ACM Trans. on Computational Biology and Bioinformatics*, vol. 8, no. 4, pp. 943-958, 2011.

[88] D. Machado, R. Costa, M. Rocha, E. Ferreira, B. Tidor, and I. Rocha, “Modeling formalisms in systems biology,” *AMB Express*, vol. 1, no. 1, p. 45, 2011.

[89] J. Roberts, “State of the art: coordinated & multiple views in exploratory visualization,” *5th Int. Conf. on Coordinated and Multiple Views in Exploratory Visualization*, pp. 61-71, 2007.

[90] B. Shneiderman and A. Aris, “Network visualization by semantic substrates,” *IEEE Trans. on Visualization and Computer Graphics*, vol. 12, no. 5, pp. 733-740, 2006.

[91] C. Collins and S. Carpendale, “VisLink: Revealing relationships amongst visualizations,” *IEEE Trans. on Visualization and Computer Graphics*, vol. 13, no. 6, pp. 1192-1199, 2007.

[92] M. Streit, A. Lex, M. Kalkusch, K. Zatloukal, and D. Schmalstieg, “Caleydo: connecting pathways and gene expression,” *Bioinformatics*, vol. 25, no. 20, pp. 2760-2761, 2009.

[93] M. Steinberger, M. Waldner, M. Streit, A. Lex, and D. Schmalstieg, “Context-preserving visual links,” *IEEE Trans. on Visualization and Computer Graphics*, vol. 17, no. 12, pp. 2249-2258, 2011.

[94] N. Novere, M. Hucka, H. Mi, S. Moodie, F. Schreiber, *et al. Nature Biotechnology*, vol. 27, no. 8, pp. 735-741, 2009.

[95] R. Sharan and T. Ideker, “Modeling cellular machinery through biological network comparison,” *Nature Biotechnology*, vol. 24, no. 4, pp. 427-433, 2006.

[96] H. Ogata, W. Fujibuchi, S. Goto, and M. Kanehisa, "A heuristic graph comparison algorithm and its application to detect functionally related enzyme clusters," *Nucleic Acids Research*, vol. 28, no. 20, pp. 4021-4028, 2000.

[97] J. Berg and M. Lässig, "Local graph alignment and motif search in biological networks," *Proc. of the National Academy of Sciences*, vol. 101, no. 41, pp. 14689-14694, 2004.

[98] N. Pržulj, "Biological network comparison using graphlet degree distribution," *Bioinformatics*, vol. 23, no. 2, pp. e177-e183, 2006.

[99] G. W. Klau, "A new graph-based method for pairwise global network alignment," *BMC Bioinformatics*, vol. 19, no. Suppl 1, p. S59, 2009.

[100] R. Pinter, O. Rokhlenko, E. Yeger-Lotem, and M. Ziv-Ukelson, "Alignment of metabolic pathways," *Bioinformatics*, vol. 21, no. 16, pp. 3401-3408, 2005.

[101] M. Koyutürk, Y. Kim, S. Subramaniam, W. Szpankowski, and A. Grama, "Detecting conserved interaction patterns in biological networks," *Journal of Computational Biology*, vol. 13, no. 7, pp. 1299-1322, 2006.

[102] R. Sharan, S. Suthram, R. Kelley, T. Kuhn, S. McCuine, P. Uetz, T. Sittler, R. Karp, and T. Ideker, "Conserved patterns of protein interaction in multiple species," *Proc. of the National Academy of Sciences*, vol. 102, no. 6, pp. 1974-1979, 2005.

[103] S. Brasch, L. Linsen, and G. Fuellen, "Visualization of aligned biological networks: A survey," *Int. Conf. on Cyberworlds*, pp. 49-53, 2007.

[104] D. Fung, S.-H. Hong, D. Koschützki, F. Schreiber, and K. Xu, "2.5D visualization of overlapping biological networks," *Journal of Integrative Bioinformatics*, vol. 5, no. 1, p. 90, 2008.

[105] S. Brasch, L. Linsen, and G. Fuellen, "VANLO - interactive visual exploration of aligned biological networks," *BMC Bioinformatics*, vol. 10, no. 1, p. 327, 2009.

[106] K. Andrews, M. Wohlfahrt, and G. Wurzinger, "Visual graph comparison," *13th Int. Conf. Information Visualization*, pp. 62-67, 2009.

[107] M. Meyer, B. Wong, M. Styczynski, T. Munzner, and H. Pfister, "Pathline: A tool for comparative functional genomics," *Computer Graphics Forum*, vol. 29, no. 3, pp. 1043-1052, 2010.

[108] M. Ghoniem, J.-D. Fekete, and P. Castagliola, "A comparison of the readability of graphs using node-link and matrix-based representations," *IEEE Symp. on Information Visualization*, pp. 17-24, 2004.

[109] J. Abello and F. van Ham, “Matrix Zoom: A visual interface to semi-external graphs,” *IEEE Symp. on Information Visualization*, pp. 183-190, 2004.

[110] F. van Ham, “Using multilevel call matrices in large software projects,” *IEEE Symp. on Information Visualization*, pp. 227-232, 2003.

[111] W. Didimo, G. Liotta, and S. Romeo, “Topology-driven force-directed algorithms,” in *Graph Drawing*, vol. 6502 of *Lecture Notes in Computer Science*, pp. 165-176, 2011.

[112] T. Dwyer, Y. Koren, and K. Marriott, “Ipsep-cola: An incremental procedure for separation constraint layout of graphs,” *IEEE Trans. on Visualization and Computer Graphics*, vol. 12, no. 5, pp. 821-828, 2006.

[113] P. Simonetto, D. Archambault, D. Auber, and R. Bourqui, “ImPrEd: An improved force-directed algorithm that prevents nodes from crossing edges,” *Computer Graphics Forum*, vol. 30, no. 3, pp. 1071-1080, 2011.

[114] E. Reingold and J. Tilford, “Tidier drawings of trees,” *IEEE Trans. on Software Engineering*, vol. SE-7, no. 2, pp. 223-228, 1981.

[115] D. Croft, G. O’Kelly, G. Wu, R. Haw, M. Gillespie, L. Matthews, M. Caudy, P. Garapati, G. Gopinath, B. Jassal, S. Jupe, I. Kalatskaya, S. Mahajan, B. May, N. Ndegwa, E. Schmidt, V. Shamovsky, C. Yung, E. Birney, H. Hermjakob, P. D’Eustachio, and L. Stein, “Reactome: A database of reactions, pathways and biological processes,” *Nucleic Acids Research*, vol. 39, no. suppl 1, pp. D691-D697, 2011.

[116] L. Matthews, G. Gopinath, M. Gillespie, M. Caudy, D. Croft, B. de Bono, P. Garapati, J. Hemish, H. Hermjakob, B. Jassal, A. Kanapin, S. Lewis, S. Mahajan, B. May, E. Schmidt, I. Vastrik, G. Wu, E. Birney, L. Stein, and P. D’Eustachio, “Reactome knowledgebase of human biological pathways and processes,” *Nucleic Acids Research*, vol. 37, no. suppl 1, pp. D619-D622, 2009.

[117] M. Graham, J. Kennedy, T. Paterson, and A. Law, “Visualising errors in animal pedigree genotype data,” *Computer Graphics Forum*, vol. 30, no. 3, pp. 1011-1020, 2011.

[118] D. Harel and Y. Koren, “A fast multi-scale method for drawing large graphs,” in *Graph Drawing*, vol. 1984 of *Lecture Notes in Computer Science*, pp. 183-196, 2001.

[119] M. Graham and J. Kennedy, “Exploring multiple trees through DAG representations,” *IEEE Trans. on Visualization and Computer Graphics*, vol. 13, no. 6, pp. 1294-1301, 2007.

[120] N. Elmqvist, T. Do, H. Goodell, N. Henry, and J.-D. Fekete, “Zame: Interactive large-scale graph visualization,” *IEEE Pacific Visualization Symp.*, pp. 215-222, 2008.

[121] D. Archambault, T. Munzner, and D. Auber, “GrouseFlocks: Steerable exploration of graph hierarchy space,” *IEEE Trans. on Visualization and Computer Graphics*, vol. 14, no. 4, pp. 900-913, 2008.

[122] Y. Makita, M. Nakao, N. Ogasawara, and K. Nakai, “DBTBS: Database of transcriptional regulation in *Bacillus subtilis* and its contribution to comparative genomics,” *Nucleic Acids Research*, vol. 32, no. suppl 1, pp. D75-D77, 2004.

[123] H. Salgado, S. Gama-Castro, M. Peralta-Gil, E. Díaz-Peredo, F. Sánchez-Solano, A. Santos-Zavaleta, I. Martínez-Flores, V. Jiménez-Jacinto, C. Bonavides-Martínez, J. Segura-Salazar, A. Martínez-Antonio, J. Collado-Vides, and J. Collado-Vides, “RegulonDB (version 5.0): K-12 transcriptional regulatory network, operon organization, and growth conditions,” *Nucleic Acids Research*, vol. 34, no. suppl 1, pp. D394-D397, 2006.

[124] K. Boitmanis, U. Brandes, and C. Pich, “Visualizing internet evolution on the autonomous systems level,” in *Graph Drawing*, vol. 4875 of *Lecture Notes in Computer Science*, pp. 365-376, 2008.

[125] R. Andersen, F. Chung, and L. Lu, “Drawing power law graphs using a local/global decomposition,” *Algorithmica*, vol. 47, no. 4, pp. 379-397, 2007.

[126] C. Collins, G. Penn, and S. Carpendale, “Bubble Sets: Revealing set relations with isocontours over existing visualizations,” *IEEE Trans. on Visualization and Computer Graphics*, vol. 15, no. 6, pp. 1009-1016, 2009.

[127] B. Alper, N. Riche, G. Ramos, and M. Czerwinski, “Design study of LineSets, a novel set visualization technique,” *IEEE Trans. on Visualization and Computer Graphics*, vol. 17, no. 12, pp. 2259-2267, 2011.

[128] D. Johnson and H. Pollak, “Hypergraph planarity and the complexity of drawing Venn diagrams,” *Journal of Graph Theory*, vol. 11, no. 3, pp. 309-325, 1987.

[129] M. Kaufmann, M. van Kreveld, and B. Speckmann, “Subdivision drawings of hypergraphs,” in *Graph Drawing*, vol. 5417 of *Lecture Notes in Computer Science*, pp. 396-407, 2009.

[130] U. Brandes, S. Cornelsen, B. Pampel, and A. Sallaberry, “Blocks of hypergraphs,” in *Combinatorial Algorithms*, vol. 6460 of *Lecture Notes in Computer Science*, pp. 201-211, 2011.

[131] U. Brandes, S. Cornelsen, B. Pampel, and A. Sallaberry, “Path-based supports for hypergraphs,” in *Combinatorial Algorithms*, vol. 6460 of *Lecture Notes in Computer Science*, pp. 20-33, 2011.

[132] K. Buchin, M. van Kreveld, H. Meijer, B. Speckmann, and K. Verbeek, “On planar supports for hypergraphs,” in *Graph Drawing*, vol. 5849 of *Lecture Notes in Computer Science*, pp. 345-356, 2010.

[133] J. Flower, P. Rodgers, and P. Mutton, “Layout metrics for Euler diagrams,” *Int. Conf. on Information Visualization*, pp. 272-280, 2003.

[134] F. Bertault and P. Eades, “Drawing hypergraphs in the subset standard (short demo paper),” in *Graph Drawing*, vol. 1984 of *Lecture Notes in Computer Science*, pp. 164-169, 2001.

[135] P. Simonetto, D. Auber, and D. Archambault, “Fully automatic visualisation of overlapping sets,” *Computer Graphics Forum*, vol. 28, no. 3, pp. 967-974, 2009.

[136] N. Henry Riche and T. Dwyer, “Untangling Euler diagrams,” *IEEE Trans. on Visualization and Computer Graphics*, vol. 16, no. 6, pp. 1090-1099, 2010.

[137] R. Santamaría and R. Therón, “Visualization of intersecting groups based on hypergraphs,” *IEICE Trans. on Information and Systems*, vol. 93-D, no. 7, pp. 1957-1964, 2010.

[138] W. Freiler, K. Matkovic, and H. Hauser, “Interactive visual analysis of set-typed data,” *IEEE Trans. on Visualization and Computer Graphics*, vol. 14, no. 6, pp. 1340-1347, 2008.

[139] H. Byelas and A. Telea, “Visualization of areas of interest in software architecture diagrams,” *Proc. of the ACM Symp. on Software Visualization*, pp. 105-114, 2006.

[140] H. Byelas and A. Telea, “Towards realism in drawing areas of interest on architecture diagrams,” *Journal of Visual Languages and Computing*, vol. 20, no. 2, pp. 110-128, 2009.

[141] S. van Dijk, M. van Kreveld, T. Strijk, and A. Wolff, “Towards an evaluation of quality for names placement methods,” *International Journal of Geographical Information Science*, vol. 16, no. 7, pp. 641-661, 2002.

[142] R. Wein, J. van den Berg, and D. Halperin, “The visibility-Voronoi complex and its applications,” *Computational Geometry*, vol. 36, no. 1, pp. 66-87, 2007.

[143] M. de Berg, O. Cheong, M. van Kreveld, and M. Overmars, *Computational Geometry: Algorithms and Applications*. Springer, 2008.

[144] B. Dent, J. Torguson, and T. Hodler, *Cartography: Thematic Map Design*. McGraw-Hill, 2008.

[145] Vivid Solutions, "Java Topology Suite." www.vividsolutions.com/jts.

[146] J. Krause, "Bubble sets library." github.com/JosuaKrause/Bubble-Sets.

[147] W. Meulemans, N. Henry Riche, B. Speckmann, B. Alper, and T. Dwyer, "Kelp-Fusion: A hybrid set visualization technique," *IEEE Trans. on Visualization and Computer Graphics*, vol. 19, no. 11, pp. 1846-1858, 2013.

[148] A. Alizadeh, M. Eisen, R. Davis, C. Ma, *et al.*, "Distinct types of diffuse large B-cell lymphoma identified by gene expression profiling," *Nature*, vol. 403, no. 6769, pp. 503-511, 2000.

[149] T. Golub, D. Slonim, P. Tamayo, C. Huard, M. Gaasenbeek, J. Mesirov, H. Coller, M. Loh, J. Downing, and M. Caligiuri, "Molecular classification of cancer: class discovery and class prediction by gene expression monitoring," *Science*, vol. 286, no. 5439, pp. 531-537, 1999.

[150] M. van de Vijver, Y. He, L. van't Veer, H. Dai, *et al.*, "A gene-expression signature as a predictor of survival in breast cancer.," *The New England journal of medicine*, vol. 347, no. 25, pp. 1999-2009, 2002.

[151] A. Tarca, S. Draghici, P. Khatri, S. Hassan, P. Mittal, J. Kim, C. Kim, J. Kusanovic, and R. Romero, "A novel signaling pathway impact analysis.," *Bioinformatics*, vol. 25, no. 1, pp. 75-82, 2009.

[152] C. Vaske, S. Benz, J. Sanborn, D. Earl, C. Szeto, J. Zhu, D. Haussler, and J. Stuart, "Inference of patient-specific pathway activities from multi-dimensional cancer genomics data using PARADIGM," *Bioinformatics*, vol. 26, no. 12, pp. i237-i245, 2010.

[153] T. Ideker, O. Ozier, B. Schwikowski, and A. Siegel, "Discovering regulatory and signalling circuits in molecular interaction networks," *Bioinformatics*, vol. 18, no. Suppl 1, pp. S233-S240, 2002.

[154] M. Dittrich, G. Klau, A. Rosenwald, T. Dandekar, and T. Müller, "Identifying functional modules in protein-protein interaction networks: an integrated exact approach," *Bioinformatics*, vol. 24, no. 13, pp. i223-i231, 2008.

[155] M. Kanehisa and S. Goto, "KEGG: Kyoto encyclopedia of genes and genomes," *Nucleic Acids Research*, vol. 28, no. 1, pp. 27-30, 2000.

[156] K. Mitra, A.-R. Carvunis, S. K. Ramesh, and T. Ideker, "Integrative approaches for finding modular structure in biological networks," *Nature Reviews Genetics*, vol. 14, no. 10, pp. 719-732, 2013.

[157] M. Ashburner, C. Ball, J. Blake, D. Botstein, H. Butler, M. Cherry, A. Davis, K. Dolinski, S. Dwight, J. Eppig, *et al.*, "Gene Ontology: tool for the unification of biology," *Nature Genetics*, vol. 25, no. 1, pp. 25-29, 2000.

[158] N. Gehlenborg, S. O'Donoghue, N. Baliga, A. Goesmann, M. Hibbs, H. Kitano, O. Kohlbacher, H. Neuweger, R. Schneider, D. Tenenbaum, and A.-C. Gavin, "Visualization of omics data for systems biology," *Nature Methods*, vol. 7, no. 3s, pp. S56-S68, 2010.

[159] M. van Iersel, T. Kelder, A. Pico, K. Hanspers, S. Coort, B. Conklin, and C. Evelo, "Presenting and exploring biological pathways with PathVisio," *BMC Bioinformatics*, vol. 9, no. 1, p. 399, 2008.

[160] "Venn and Euler Diagrams." apps.cytoscape.org/apps/vennandeulerdiagrams.

[161] "RBVI Cytoscape Plugins - Cytoscape Group Support." www.rbvi.ucsf.edu/cytoscape/groups.

[162] A. Smith, W. Xu, Y. Sun, J. Faeder, and G. Marai, "Rulebender: integrated modeling, simulation and visualization for rule-based intracellular biochemistry," *BMC bioinformatics*, vol. 13, no. Suppl 8, p. S3, 2012.

[163] K. Sugiyama and K. Misue, "Visualization of structural information: automatic drawing of compound digraphs," *IEEE Trans. on Systems, Man and Cybernetics*, vol. 21, no. 4, pp. 876-892, 1991.

[164] T. Dwyer, K. Marriott, F. Schreiber, P. Stuckey, M. Woodward, and M. Wybrow, "Exploration of networks using overview+detail with constraint-based cooperative layout," *IEEE Trans. on Visualization and Computer Graphics*, vol. 14, no. 6, pp. 1293-1300, 2008.

[165] E. Gansner, Y. Hu, and S. Kobourov, "Gmap: Visualizing graphs and clusters as maps," *IEEE Pacific Visualization Symp.*, pp. 201-208, 2010.

[166] T. Dwyer, K. Marriott, and P. Stuckey, "Fast node overlap removal," in *Graph Drawing*, vol. 3843 of *Lecture Notes in Computer Science*, pp. 153-164, 2006.

[167] T. Kohonen, "The self-organizing map," *Proc. of the IEEE*, vol. 78, no. 9, pp. 1464-1480, 1990.

[168] J. Vesanto, "SOM-based data visualization methods," *Intelligent Data Analysis*, vol. 3, no. 2, pp. 111-126, 1999.

[169] R. MacCallum, S. Redmond, and G. Christophides, "An expression map for anopheles gambiae," *BMC Genomics*, vol. 12, no. 1, p. 620, 2011.

[170] Y. Frishman and A. Tal, "Online dynamic graph drawing," *IEEE Trans. on Visualization and Computer Graphics*, vol. 14, no. 4, pp. 727-740, 2008.

[171] M. Harrower and C. Brewer, "Colorbrewer.org: An online tool for selecting colour schemes for maps," *The Cartographic Journal*, vol. 40, no. 1, pp. 27-37, 2003.

[172] M. Gandhi and R. Khanna, "Human cytomegalovirus: clinical aspects, immune regulation, and emerging treatments," *The Lancet Infectious Diseases*, vol. 4, no. 12, pp. 725-738, 2004.

[173] C. Söderberg-Nauclér, "Does cytomegalovirus play a causative role in the development of various inflammatory diseases and cancer?," *Journal of Internal Medicine*, vol. 259, no. 3, pp. 219-246, 2006.

[174] C. Cobbs, L. Harkins, M. Samanta, Y. Gillespie, S. Bharara, P. King, L. B. Nabors, C. Cobbs, and W. Britt, "Human cytomegalovirus infection and expression in human malignant glioma," *Cancer Research*, vol. 62, no. 12, pp. 3347-3350, 2002.

[175] L. Harkins, A. Volk, M. Samanta, I. Mikolaenko, W. Britt, K. Bland, and C. Cobbs, "Specific localisation of human cytomegalovirus nucleic acids and proteins in human colorectal cancer," *The Lancet*, vol. 360, no. 9345, pp. 1557-1563, 2002.

[176] J. Cinatl, J. Vogel, R. Kotchetkov, and H. Doerr, "Oncomodulatory signals by regulatory proteins encoded by human cytomegalovirus: a novel role for viral infection in tumor progression," *FEMS Microbiology Reviews*, vol. 28, pp. 59-77, 2004.

[177] J. Randolph-Habecker, B. Rahill, B. Torok-Storb, J. Vieira, P. Kolattukudy, B. Rovin, and D. Sedmak, "The expression of the cytomegalovirus chemokine receptor homolog US28 sequesters biologically active CC chemokines and alters IL-8 production," *Cytokine*, vol. 19, no. 1, pp. 37-46, 2002.

[178] P. Casarosa, R. Bakker, D. Verzijl, M. Navis, H. Timmerman, R. Leurs, and M. Smit, "Constitutive signaling of the human cytomegalovirus-encoded chemokine receptor US28," *Journal of Biological Chemistry*, vol. 276, no. 2, pp. 1133-1137, 2001.

[179] D. Maussang, D. Verzijl, M. van Walsum, R. Leurs, J. Holl, O. Pleskoff, D. Michel, G. van Dongen, and M. Smit, “Human cytomegalovirus-encoded chemokine receptor US28 promotes tumorigenesis,” *Proc. of the National Academy of Sciences*, vol. 103, no. 35, pp. 13068-13073, 2006.

[180] D. Maussang, E. Langemeijer, C. Fitzsimons, M. Stigter-van Walsum, R. Dijkman, M. Borg, E. Slinger, A. Schreiber, D. Michel, C. Tensen, G. van Dongen, R. Leurs, and M. Smit, “The human cytomegalovirus-encoded chemokine receptor US28 promotes angiogenesis and tumor formation via cyclooxygenase-2,” *Cancer Research*, vol. 69, no. 7, pp. 2861-2869, 2009.

[181] E. Slinger, D. Maussang, A. Schreiber, M. Siderius, A. Rahbar, A. Fraile-Ramos, S. Lira, C. Soderberg-Naucler, and M. Smit, “HCMV-encoded chemokine receptor US28 mediates proliferative signaling through the IL-6-STAT3 axis,” *Science Signaling*, vol. 3, no. 133, p. ra58, 2010.

[182] E. Langemeijer, E. Slinger, S. de Munnik, A. Schreiber, D. Maussang, H. Vischer, F. Verkaar, R. Leurs, M. Siderius, and M. Smit, “Constitutive β-catenin signaling by the viral chemokine receptor US28,” *PLoS ONE*, vol. 7, no. 11, p. e48935, 2012.

[183] L. Gautier, L. Cope, B. Bolstad, and R. Irizarry, “affy - analysis of affymetrix genechip data at the probe level,” *Bioinformatics*, vol. 20, no. 3, pp. 307-315, 2004.

[184] G. Smyth, “Limma: linear models for microarray data,” *Bioinformatics and Computational Biology Solutions Using R and Bioconductor*, pp. 397-420, 2005.

[185] A. Alexa, J. Rahnenfuhrer, and T. Lengauer, “Improved scoring of functional groups from gene expression data by decorrelating GO graph structure,” *Bioinformatics*, vol. 22, no. 13, pp. 1600-1607, 2006.

[186] R. Minisini, C. Tulone, A. Lüske, D. Michel, T. Mertens, P. Gierschik, and B. Moepps, “Constitutive inositol phosphate formation in cytomegalovirus-infected human fibroblasts is due to expression of the chemokine receptor homologue pUS28,” *Journal of Virology*, vol. 77, no. 8, pp. 4489-4501, 2003.

[187] J. Zhurinsky, M. Shtutman, and A. Ben-Ze’ev, “Differential mechanisms of LEF/TCF family-dependent transcriptional activation by β-catenin and plakoglobin,” *Molecular and Cellular Biology*, vol. 20, no. 12, pp. 4238-4252, 2000.

[188] D. Streblow, C. Soderberg-Naucler, J. Vieira, P. Smith, E. Wakabayashi, F. Ruchti, K. Mattison, Y. Altschuler, and J. Nelson, “The human cytomegalovirus chemokine receptor US28 mediates vascular smooth muscle cell migration,” *Cell*, vol. 99, no. 5, pp. 511-520, 1999.

[189] D. Streblow, J. Vomaske, P. Smith, R. Melnychuk, L. Hall, D. Pancheva, M. Smit, P. Casarosa, D. Schlaepfer, and J. Nelson, “Human cytomegalovirus chemokine receptor US28-induced smooth muscle cell migration is mediated by focal adhesion kinase and Src,” *Journal of Biological Chemistry*, vol. 278, no. 50, pp. 50456-50465, 2003.

[190] M. Herynk, R. Tsan, R. Radinsky, and G. Gallick, “Activation of c-Met in colorectal carcinoma cells leads to constitutive association of tyrosine-phosphorylated β-catenin,” *Clinical & Experimental Metastasis*, vol. 20, no. 4, pp. 291-300, 2003.

[191] R. Purcell, M. Childs, R. Maibach, C. Miles, C. Turner, A. Zimmermann, and M. Sullivan, “HGF/c-Met related activation of beta-catenin in hepatoblastoma,” *Journal of Experimental & Clinical Cancer Research*, vol. 30, no. 1, p. 96, 2011.

[192] Y.-Y. Ahn, J. Bagrow, and S. Lehmann, “Link communities reveal multiscale complexity in networks,” *Nature*, vol. 466, no. 7307, pp. 761-764, 2010.

[193] S. Hadlak, H. Schumann, C. Cap, and T. Wollenberg, “Supporting the visual analysis of dynamic networks by clustering associated temporal attributes,” *IEEE Trans. on Visualization and Computer Graphics*, vol. 19, no. 12, pp. 2267-2276, 2013.

[194] S. van den Elzen and J. van Wijk, “Multivariate network exploration and presentation: From detail to overview via selections and aggregations,” *IEEE Trans. on Visualization and Computer Graphics*, vol. 20, no. 12, pp. 2310-2319, 2014.

[195] C. Vehlow, J. Hasenauer, F. Theis, and D. Weiskopf, “Visualizing edge-edge relations in graphs,” *IEEE Pacific Visualization Symp.*, vol. 2013, pp. 201-208, 2013.

[196] C. Vehlow, T. Reinhardt, and D. Weiskopf, “Visualizing fuzzy overlapping communities in networks,” *IEEE Trans. on Visualization and Computer Graphics*, vol. 19, no. 12, pp. 2486-2495, 2013.

[197] B. Adamcsek, G. Palla, I. Farkas, I. Derényi, and T. Vicsek, “Cfinder: locating cliques and overlapping modules in biological networks,” *Bioinformatics*, vol. 22, no. 8, pp. 1021-1023, 2006.

[198] T. Itoh, C. Muelder, K.-L. Ma, and J. Sese, “A hybrid space-filling and force-directed layout method for visualizing multiple-category graphs,” *IEEE Pacific Visualization Symp.*, pp. 121-128, 2009.

[199] R. Kosara, F. Bendix, and H. Hauser, “Parallel sets: interactive exploration and visual analysis of categorical data,” *IEEE Trans. on Visualization and Computer Graphics*, vol. 12, no. 4, pp. 558-568, 2006.

[200] B. Alsallakh, W. Aigner, S. Miksch, and H. Hauser, “Radial sets: Interactive visual analysis of large overlapping sets,” *IEEE Trans. on Visualization and Computer Graphics*, vol. 19, pp. 2496-2505, 2013.

[201] R. Sadana, T. Major, A. Dove, and J. Stasko, “Onset: A visualization technique for large-scale binary set data,” *IEEE Trans. on Visualization and Computer Graphics*, vol. 20, no. 12, pp. 1993-2002, 2014.

[202] A. Kerren and I. Jusufi, “A Novel Radial Visualization Approach for Undirected Hypergraphs,” *Eurographics Conf. on Visualization - Short Papers*, 2013.

[203] A. Lex, N. Gehlenborg, H. Strobelt, R. Vuillemot, and H. Pfister, “Upset: Visualization of intersecting sets,” *IEEE Trans. on Visualization and Computer Graphics*, vol. 20, no. 12, pp. 1983-1992, 2014.

[204] B. Alsallakh, L. Micallef, W. Aigner, H. Hauser, S. Miksch, and P. Rodgers, “Visualizing sets and set-typed data: State-of-the-art and future challenges,” *Eurographics Conf. on Visualization*, 2014.

[205] B. Saket, P. Simonetto, and S. Kobourov, “Group-Level Graph Visualization Taxonomy,” *Eurographics Conf. on Visualization - Short Papers*, 2014.

[206] Z. Liu, S. Navathe, and J. Stasko, “Network-based visual analysis of tabular data,” *IEEE Conf. on Visual Analytics Science and Technology*, pp. 41-50, Oct 2011.

[207] A. Lambert, F. Queyroi, and R. Bourqui, “Visualizing patterns in node-link diagrams,” *Int. Conf. on Information Visualisation*, pp. 48-53, 2012.

[208] T. Madhyastha and T. Grabowski, “Age-related differences in the dynamic architecture of intrinsic networks,” *Brain Connectivity*, vol. 4, no. 4, pp. 231-241, 2014.

List of Publications

[209] K. Dinkla, M. Westenberg, H. Timmerman, S. van Hijum, and J. van Wijk, “Comparison of multiple weighted hierarchies: Visual analytics for microbe community profiling,” *Computer Graphics Forum*, vol. 30, no. 3, pp. 1141-1150, 2011.

[210] K. Dinkla and M. Westenberg, “Network visualization in cell biology,” *Tsinghua Science and Technology*, vol. 17, no. 4, pp. 365-382, 2012.

[211] K. Dinkla, M. Westenberg, and J. van Wijk, “Compressed adjacency matrices: Untangling gene regulatory networks,” *IEEE Trans. on Visualization and Computer Graphics*, vol. 18, no. 12, pp. 2457-2466, 2012.

[212] K. Dinkla, M. van Kreveld, B. Speckmann, and M. Westenberg, “Kelp diagrams: Point set membership visualization,” *Computer Graphics Forum*, vol. 31, no. 3, pp. 875-884, 2012.

[213] K. Dinkla, M. El-Kebir, C.-I. Bucur, M. Siderius, M. Smit, M. Westenberg, and G. Klau, “eXamine: Exploring annotated modules in networks,” *BMC Bioinformatics*, vol. 15, no. 1, p. 201, 2014. K. Dinkla and M. El-Kebir share first authorship.

[214] K. Dinkla, N. Riche, and M. Westenberg, “Dual adjacency matrix: Exploring link groups in dense networks,” *Computer Graphics Forum*, vol. 34, no. 3, 2015.

Summary

Visualization of Modular Structures in Biological Networks

Many of the processes known to take place in biological cells are analyzed in the form of different types of network. Our knowledge of cell processes increases, but the size and complexity of these networks impedes their analysis. An important means for the bite-size analysis and understanding of biological networks is the distillation of their modular structures. We introduce, demonstrate, and discuss several visualization techniques that support biological network analysis by leveraging modular structures:

Compressed Adjacency Matrices Gene regulatory networks have structural characteristics that impede standard visualization techniques such as node-link diagrams and adjacency matrices. However, we show how these characteristics can be leveraged to cut open, rearrange, and compress adjacency matrices. These compressed matrices enable easy, visual detection of modular structures commonly associated with specific regulatory functions.

Kelp Diagrams Some situations merit explicit emphasis of modular structures, such as highlighting a region of interest on top of an existing metabolic pathway visualization. When dealing with a pre-positioned node-link diagram, delimiting structures with contours is an obvious approach but one that requires care. We introduce Kelp Diagrams to emphasize regions of interest on pre-positioned elements.

eXamine Biologists are specialized in specific cell processes, types, and conditions. Biologists therefore analyze only small modules within larger Protein Interaction Networks. Commonly, a multitude of set-based annotations are attached to such a module via readily available knowledge such as gene ontology terms. We therefore present eXamine, which is a Cytoscape app that supports the analysis of annotated modules. It integrates Kelp Diagram methodology with a simple layout algorithm and enables interactive exploration of a module and its annotations.

Dual Adjacency Matrices Modular structures in networks are often interpreted as a group of nodes accompanied by a group of links that connect them, also known as an induced subnetwork. We explore the alternate perspective; grouped links that are connected by nodes. In doing so we introduce the Dual Adjacency Matrix, which combines link and node group techniques such that an analyst can navigate the conceptual gap between link and node groups.

Curriculum Vitæ

13-07-1987	Born in Eindhoven, the Netherlands
1991-1999	Grammar School
1999-2004	General Secondary Education (HAVO)
2004-2005	Computer Software Engineering Fontys Hogescholen Eindhoven
2005-2008	Bachelor in Computer Science Eindhoven University of Technology Graduated Cum Laude
2008-2010	Master in Computer Science Eindhoven University of Technology Thesis: Visual Analytics for Microbial Phylogeny Supervised by M.A. Westenberg Graduated Cum Laude
2010-2015	Doctoral Candidate in Computer Science Eindhoven University of Technology Thesis: Visualization of Modular Structures in Biological Networks Supervised by M.A. Westenberg and J.J. van Wijk
2014	Three month internship Microsoft Research, Redmond, USA Project: Link Community Visualization Supervised by N. Henry Riche