Article

Challenges and opportunities for visualization and analysis of graph-modeled medical data

Giuseppe Agapito [1,†,‡], Pietro H Guzzi [1,‡*] and Mario Cannataro[2,]

[1] Department of Medical and Surgical Sciences; {agapito,hguzzi,cannataro}@unicz.it

* Correspondence: Pietro Hiram Guzzi hguzzi@unicz.it; Tel.: +39-0961-3694148

‡ These authors contributed equally to this work.

Abstract: Graphs are largely used in computer science to model relations, or associations, among entities the compose complex systems. More recently, they found a broad field of application in bioinformatics and medical informatics supporting modelling, analysis of many systems. The applications span from representing interactions among molecules within cells, to model the functions of the brains. In order to support research, algorithms from graph theory that are able to extract knowledge from network should be coupled to efficient visualisation techniques. Although the importance of such topic, we retain that many challenges should be faced in the futures, such as automated pathway and network layout to better match biologists' needs, the developing of a standard able to give more and better choices to represent nodes, edges finally, the developing of automated mechanism able to improve the network navigation methods that help users manage large and complex networks with the goal to improve the usability.

Keywords: Biological Network; Graphs; Statistical Analysis;

1. Introduction

Graphs although are an abstract mathematical formalism, are very useful and powerful tool to represent interactions among whatever interacting entities. Thus, graphs are the perfect tool to represent each kind of interactions that occur in a system including biological networks, computers' networks, software developing process, protein interaction networks and sub-cellular interactions. It has been shown that living organisms are extremely complex from a microscopic point of view, the whole biomolecules of an organism work together in a coordinated fashion to reach specific targets. To analyze this complex web of interactions, it is mandatory understand the roles of interactions among genes, proteins, and other cell components. To simplify the work of researchers it is necessary to develop appropriate tools and formalisms able to deal with this intricate web of interactions, making possible to represent in a graphical fashion the network under analysis as well as to get at glance an overview of the interactions. This is due to the fact that, for our brain is easier to elaborate pictures than text or number, making possible to infer some properties of the network under analysis just looking it, as described in [1].

Formally graphs are discrete mathematical object defined as a couple $G =< V, E >$ where $V = v_1, v_2, \ldots v_n$ is the set of nodes, whereas $E =< V \times V >$ is the set of edges connecting interacting nodes among them. The vertices of a graph are sometimes called nodes or dot; edges are sometimes called links, lines, arcs or connections. The set of vertices of **G** is denoted by ***V(G)*** and the set of edges is denoted by ***E(G)***. In a short form a graph can be referred as: G. Following are some basic concepts of graphs theory used as basic-tools for the analysis of graphs and networks. The number of vertices of a graph G is its *order*, written as $|G|$; instead, its number of edges is denoted by $\| G \|$. A graph of order 0 or 1 is called trivial. A vertex v is incident with an edge e if $v \in e$; then e is an edge at v. The two vertices incident with an edge are its *endvertices* or *ends*, and an edge joins its ends. An edge (x, y) is usually written as xy (or yx). Two vertices x and y of G are adjacent, or neighbours, if (x, y) is an edge of G $(xy \in E(G))$. If all the vertices of G are pairwise adjacent, then G is complete. The *degree (or valency)* $dG(v) = d(v)$ of a vertex v is the number $|E(v)|$ of edges at v; this is equal to the number of neighbours of v. A vertex of degree 0 is isolated. A *path* is a non-empty graph $P = (V, E)$ of the form $V = \{x_0, x_1, \ldots, x_k\}, E = \{x_0x_1, x_1x_2, \ldots, x_{k-1}x_k\}$, where the x_i are all distinct. The vertices x_0 and x_k are linked by P and are called its ends; the vertices $x_1, \ldots, x_{k-1}$ are the inner vertices of P . The number of edges of a path is its *length*. If every edge has not direction (joins an unordered pair of vertices) the graph is called *undirected*, this means that, it is possible walk through the edge in both direction; the edge (a, b) is the same of (b, a). Otherwise, if the edge has a direction associated the graph is called *directed graph* or *digraph*. Directed graph are employed to model *signal transduction pathways* and *gene regulation netwroks*. Furthermore, if the digraph does not contain cycles is called ***Directed Acyclic Graph*** (***DAG***). Sometimes, it is need to draw graphs where it is possible to have both edges with or without orientation. A graph extension able to represent and to manage correctly the two different types of edges is called ***Mixed Graph***. A mixed graph ***G*** consists of three finite set: the set of vertices ***V(G)***, the set of undirected edges ***E(G)*** and the set of directed edges ***A(G)***, and in short the graph can be wrote by means the following triple: $G = (V, E, A)$. Representing interactions through graphs pose several challenges depending on the context, thus the basic features of a graph should be drawn in different ways. Nodes may be drawn as dots, circles, boxes, or represented implicitly by their name labels. Edges may be drawn as straight lines, dotted lines, or bends. The information corresponding to the nodes and edges can be visualized using text labels at various positions in or next to a graph object, different colors, or other visual elements such as thickness of lines, size of boxes, etc. Moreover a graph may be drawn in the plane 2D or in three dimensions 3D. Once decided upon the representation must be faced the question of how to convey the nodes and edges of a graph. A drawing, is a mapping of the nodes and edges. But what distinguishes a good representation from a bad one? For example, drawing a *Entity Relation Diagram* with an algorithm designed to draw *Protein Protein Interaction Networks* will most likely produce a highly unsatisfactory picture. That is because an algorithm thought to draw a graph related with the protein protein interaction network also uses some general knowledge on protein interactions. Also for example, users would be highly confused if entities or relations were drawn at random locations instead of locations that highlight in a remarkable way the connections among tables into the database. On the other hand, biologists would expect that the drawing reflects the results of the interaction; in order that proteins involved in the same process are drawn next to each other for example. As well as, the need of efficient tool for graphs analysis arise. This is a good opportunities to computer scientist to provide tools and algorithms able to deal with the visualization and analysis of huge networks. The rest of the paper is structures as follows: in section 3 is presented the principal draw conventions; Section 4 describes a summary of the most used layout algorithms; Section 5 analyze the methodologies used to analyze networks; and finally section 7 concludes the paper.

2. Biological and Biomedical Datasets.

2.1. Protein Interaction Networks

The management of PPI data presents similar issues as those faced in other domains, i.e. PPI data need to be stored, exchanged, queried, and analysed. On the other hand, PPI data is represented by graphs and thus pose new issues in all phases of its management. This section discussed main phases and issues of PPI data management.

Regarding PPI data storage, main efforts were devoted to the definition of standards for data exchange such as HUPO PSI-MI, but currently PPI data are stored as large sets of binary interactions, without taking into account of such XML-based languages and related XML databases. The storage of PPI data could exploit some already developed storage systems for other graph-based data, such as the triple stores used for storing RDF data or the emerging graph databases [2]. In graph databases, schema and instances are modeled as graphs and data manipulation is expressed by graph-oriented operations. A graph database proposal for genomics is reported at http://www.xweave.com/people/mgraves/pubs/, while a proposal for biochemical pathways is reported in [3]. Moreover, a naming mechanism to identify in a unique way interactions has not been yet developed, and (binary) interactions are named by naming the interacting proteins.

Also PPI data querying could benefit from semi-structured or graph databases, in fact, as summarized below, existing PPI data offer only very simple retrieval mechanisms allowing to retrieve proteins interacting with a target protein or to build the PPI network from stored binary interactions. Current PPI databases surveyed in this paper do not offer sophisticated query mechanisms based on graph manipulation but, on the other hand, they constitute the only available structured repository for interaction data and allow an easy sharing and annotation of such data. Having in mind these limitations, the interaction databases presented so far can be categorized on the basis of: (i) the kind of the stored interactions, and (ii) the kind of data submission, querying and extraction.

A first distinction is made up on the basis of the kind of interaction, and differentiates databases storing experimental interactions, such as MINT or MIPS, from those storing predicted interactions such as OPHID. Moreover, the first class can be subdivided on the basis of the methodology used to populate database. Only IntAct accepts the submission of an interaction while DIP, BIND, and MINT report interaction mined from literature and curated by expert reviewers.

Database of predicted interactions can be categorized considering the methodology used to infer the prediction. In such a way, while OPHID and POINT map experimental interactions determined in model organisms into human interactions, IntNetDB and STRING build a probabilistic model integrating different evidences (such as gene co-expression).

Finally, databases can be distinguished on the basis of data querying and extraction. Generally databases provide a web based interface, except for OPHID that can be browsed also by using an ad hoc application. Users retrieve data by specifying a protein identifier, except for IntNetDB that requires a gene identifier, and for STRING that requires a gene identifier or a primary sequence. Finally, retrieved data can be exported as PSI-MI files or as textual format.

It should be noted that other minor databases storing interaction data exist, so the interested reader can see a a more exhaustive list of databases in the annual database issue of Nucleic Acid Research Journal [4].

2.2. Connectomics Datasets

3. Background on graph visualization

To represent effectively biological networks it is necessary to define a drawing convention. Where drawing convention, define the basic rules that the drawing must satisfy in order to be *admissible*. For example, in drawing biological pathway networks for a software application, it is possible to adopt the conventions where proteins are represented as circle, whereas biochemical reactions are represented as triangle etc. Furthermore, there is the necessity to introduce different type of edges able to represent every interaction among the different elements that compose the network. From this example, it is possible to note that, the drawing conventions of a real-life applications are very complex and involve the analysis of many details to take into account. A short list of the most used drawing conventions is the following. *Polyline drawing:* each edge is drawn as polygonal chain (see figure 1);

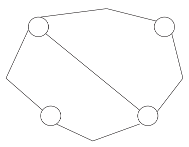

Figure 1. Representation of graph using polyline drawing.

straight line drawing: each line is drawn as a straight line (see figure 2);

orthogonal drawing: each edge is drawn as polygonal chain of alternating vertical and horizontal line (see figure 3);

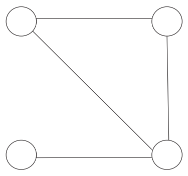

Figure 2. Representation of graph using straight line drawing.

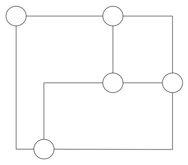

Figure 3. Representation of graph using straight line drawing.

planar drawing: drawing without crossing edges (see figure 4).

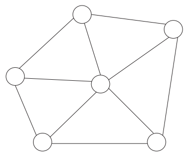

Figure 4. Representation of graph without crossing lines drawing.

The second criteria to take into account specify some properties of the drawing that would be applied as much as possible to get readability. Finding a good arrangement is thus reduced to optimizing one or a few criteria, from estetichal and technical point of view. A short list of the most used aesthetics criteria for their practical importance is: *Crossing edge minimization*. Minimization of the total number of overlapping edges. In fact, if too many edges cross each other, the human eye can not easily find out which nodes are connected by an edge. If a graph can be drawn without edge crossings (such graphs are called planar), then this is very often preferable to a drawing with edge crossings. *Bend minimization*. Minimization of the total number of bends along the edges. This is an important aesthetics criterion for orthogonal layouts because the human eye can much more easily follow an edge with none or only a few bends than an edge wildly zig-zagging through the picture. *Area minimization*. Minimization of the total area required for the draw. The ability to construct area efficient drawings is essential in practical visualization applications, where save space is a fundamental property. Moreover, a draw looks much better if the nodes and edges fill the space with homogenous density. *Symmetries*. If a graph contains symmetrical information then it is important to reflect this symmetry in its arrangement. Symmetries can be further formalized introducing a mathematical model of symmetries in graph and drawing, but unfortunately, displaying symmetries is not an easy task. *Total edge length minimization*. Minimization of the sum of the lengths of the edges. This criteria is meaningful if and only if the drawing strategy adopted prevent from being arbitrarily scaled down. *Maximum edge length minimization*. Minimization of the maximum length of an edge. This criteria is meaningful if and only if the drawing strategy adopted prevent from being arbitrarily scaled down. *Uniform edge length minimization*. Minimization of the variance of the lengths of the edges.

The third criteria of a graph drawing algorithm is related with the computational efficiency. This is a crucial aspect of graph drawing algorithm, guarantee good performance (real time response), even for large graphs. This feature become a crucial issue when, it is necessary to guarantee real time response, because it is need to make some graph analysis and allow the user to continue to interact with the graph. The aesthetics criteria are computational hard, and on the other hand are in contrast with the other criteria making very complex to deal with all them in the same time. Reason for which, there is need to define approximation strategies and heuristics to get a compromise in among the three criteria described before, in order to produce readable graphs.

4. Visualization of graph-based biomedical data

The purpose of data presentation is to reveal information buried in a set of data either to the exploration, or just in its communication. A presentation is said to be effective, if it conveys the true information in an easily comprehensible, yet precise, way. Graphs are used as abstractions in numerous fields of application. Consequently, the same structure can mean quite different things depending on the context. To effectively visualize a graph, the specific type of information it represents must govern the graphical design. At the core of any visualization tool there is a layout algorithm, i.e. an algorithm that decides how to present each object of the graph to the user. For this purpose, some layout algorithms were developed, in order to satisfy the drawing convention, some aesthetic criteria (the most useful related with the data to draw), in order to guarantee a good computational efficiency.

A brief description of the most known layout algorithms and of their features not going too far into the technical details are presented below:

- **Random Layout Algorithm [5]:** is very simple, because it arranges the graph nodes and edges in a random way on the screen. The advantage of this algorithm is its simplicity for both implementation and use, but it presents some disadvantages as high number of cross-edges produced, and a not optimum use of the available space. Random Layout is used principally to represent genetic networks .
- **Circular Layout Algorithm [6]:** how the name suggests, places the graph nodes in succession, one after the other, on a circle, so that all nodes are at the same distance from the center of the screen, It attempts to minimize overlapping vertices and cross edges (also named overlapping edges). Although this algorithm can appear trivial, it is widely used to visualize complexes and pathways .
- **Hierarchical Layout Algorithm (HLA) [7]:** HLA was invented by Sugiyama [8] and it arranges the graph nodes in different hierarchical groups with the goal to reduce the number of cross edges. A drawback is related with the effort to minimize the number of overlapping edges. Used to dynamically visualize the complex data related to pathways .
- **Fruchterman&Reingold and Eades&Peter Algorithms.** Fruchterman&Reingold algorithm [9] and Eades&Peter algorithm [10], also known as Spring Embedded algorithm, belong to the class of Force-directed algorithms. They consider each node as an electrically charged element and each edge as a spring linking two nodes. In this system, nodes with the same charge repel each other, while opposites attract, with an attraction/repulse force due to the springs. The computation time required to obtain the convergence to equilibrium grows relatively quickly with the size of the graph (that is, the number of nodes and edges) and the layout process becomes time-consuming for large graphs.
- **Kamada-Kawai Algorithm [11].** The algorithm designed by T. Kamada and S. Kawai is based on the concept of theoretic distance between the nodes. In the Kamada-Kawai algorithm the forces are represented as spring forces that are directly proportional to the graph theoretic distances.
- **Tree Layout Algorithm [12]** arranges graphs as a tree without cycles, with a hierarchical organization of the nodes, in order that the displaying is clear and easy to understand. The main advantage of the Tree Layout algorithm is its low complexity (generally, it has linear performance in the number of nodes both for binary and n-ary trees) how demonstrated in that makes it efficient and scalable, even with large graphs, but it presents a problem when arranging a huge number of nodes in limited areas as i.e. the monitor.
- **Cluster Layout Algorithm [13].** The Cluster Layout algorithm reduces the number of visible elements in order to reduce the visual complexity of a graph. Limiting the number of visible elements allows to improve the clarity and simultaneously increases performance of layout and rendering .

- **Grid Layout Algorithm.** It disposes the graph nodes on a 2-dimensional squared grid, where each node into the grid is modeled as a particle that interacts with each other. Grid Layout algorithm shapes the graph as a system of interacting particles on a grid. The particles (nodes) interact according to a cost function which is designed based on the topological structure of the network. In such a system, closely related nodes attract each other, and remotely related nodes repulse each other. In [14], an algorithm to draw complex biochemical networks is proposed. A network is modeled as a system of interacting nodes on squared grids, where a comprehensive visual representations of the network helps researchers to gain insight into the complex network data.

5. Analysis of graph-based biomedical data

As starting point, the global topology of an interaction network, i.e. the study of main properties such as the clustering coefficient or of the diameter, can reveal main properties of the network. In addition to analysis of global properties, the study of recurring local topological features and the extraction of relevant modules, i.e. cliques, may extract relevant knowledge. Finally, comparison of networks enforce the research capabilities by enabling for instance the comparison of networks corresponding to different states (e.g. healthy or diseased).

For the purposes of this work, we categorize these studies in two main classes: (i) algorithms that analyse properties of the single networks (e.g. centrality measure, community discovering), and (iii) algorithms for the comparison of two or more networks.

For instance, ethods belonging to the first class try to extract highly connected regions in a graph, under the hypothesis that they could encode biologically relevant subsystems. Algorithms belonging to the second class investigate conservation and divergence of interactions between different species or states [15,16], therefore they usually receive in input two or more PPI networks (i.e. two or more graphs) and produce as output a set of conserved subgraphs among them.

5.1. Measures

Graphs become increasingly important in modelling complicated structures, such as social networks, network of diseases, chemical compounds, protein structures, biological networks, but on the other hand there is the need to develop efficient analysis methodologies able to deal with networks [17]. Analyzing network properties may provide useful insight in the internal organization of the biological elements, molecules and macromolecules [18]. The main properties commonly analyzed in networks are the following: *Graph density* is used to evaluate how sparse or dense is a graph and it is defined as: $density = \frac{2|E|}{|V|(|V|-1)}$. *Centralization* is a measure to evaluate the average connectivity of each node of the network. Values of centralization close to 1, suggest that the network could have a star topology. Conversely for values of centralization closer to 0, show that the nodes of the network have on average the same connectivity. Centralization is obtained as: $Centralization = \frac{n}{n-2}\left(\frac{max(k)}{n-1} - densisty\right)$ Another useful properties to evaluate in biological networks is the detection of central nodes, since they play a crucial role in regulation of many biological functions. Degree Centrality shows the number of edges incidents of each single node defined as $C_D = d(v)$. In oriented graph each node present two degree centralities *in* and *out*. Nodes that present high values of C_D are called **Hub** that playing a crucial role in network topology. *Closeness Centrality* highlight relevant nodes in the network that can communicate quickly with other nodes. The closeness centrality is defined as: $CC(v) = \frac{1}{\sum_{i\in V}^{|V|} dist(v,w)}$. where $dist(v,w)$ represents the shortest path between the nodes v and w. *Betweenness Centrality* highlights relevant nodes that belong to a high number of paths between other nodes in the network. Betweenness Centrality is computed as: $BC(v) = \sum_{(i,j)\in V(v)} \frac{\sigma_{ij}(v)}{\sigma_{ij}}$. Nodes with high values of betweenness centrality are defined as the regulators of the dynamics properties.

5.2. Comparing Graphs

The comparison of biological and biomedical networks over such a framework enables researcher to distinguish recurrent patterns that represent biological functions that are conserved during the evolution [19]. In this way the search for conserved motifs in two networks can be formulated as the search for similar subgraphs in two or more networks [15,20,21]. This definition permits the definition of a *graph alignment* as counterpart of sequence alignment based on an appropriate scoring function that measures the similarities of two subgraphs enabling the global comparison of through subgraphs by the measurement of mutually similar subgraphs. Such a procedure produces as output an *alignment graph* A_l=V_a, E_a starting from two graphs G_1 and G_2. Such graph contains a node $a \in A_l$ for each pair of corresponding nodes in G_1 and G_2, while the presence of an edge $e = (a_1, a_2) \in E_a$ can be the results of a presence of a pair in both G_1 and G_2 among nodes that are collapsed in a_1 and a_2, or in only one. From an algorithmic perspective this problem is not simple because all its formulations rely on a variation of the well known subgraph isomorphism problem that is NP-complete. Thus, all the existing methods are based on some heuristics to reduce the complexity of the problem.

5.3. PPI Databases

To give an idea of the dimensions of some real complete interaction networks, in Table 1 we report the number of known protein interactions related respectively to the *Homo Sapiens*, *Mus Musculus*, and *Saccharomyces Cerevisiae* organisms. For each organism, the table reports the number of proteins (#Nodes) and the number of interactions (#Edges) of the interaction network, extracted from the DIP[1], IntAct[2], MINT[3], I2D[4], and BioGrid[5] protein interaction databases. Due to the high number of nodes and edges, it is clear that such networks need for a proper visualization tool in order to analyze effectively and quickly the enormous amount of information available. Finally, it could be noted that each database reports a different number of proteins and interactions, probably due to the curation process of such databases.

Table 1. The following table shows the number of known protein interactions related respectively to the *Homo Sapiens*, *Mus Musculus*, and *Saccharomyces Cerevisiae* organisms. For each organism, the table reports the number of proteins (#Nodes) and the number of interactions (#Edges) of the interaction network, extracted from the DIP, IntAct, MINT, I2D, and BioGrid protein interaction databases. Due to the high number of nodes and edges, it is clear that such networks need for a proper visualization tool in order to analyze effectively and quickly the enormous amount of information available. Finally, it could be noted that each database reports a different number of proteins and interactions, probably due to the curation process of such databases.

	Organism					
	Homo Sapiens		**Mus Musculus**		**Saccharomyces C**	
Databases	*#Nodes*	*#Edges*	*#Nodes*	*#Edges*	*#Nodes*	*#Edges*
I2D	14847	156188	12818	145119	11194	152877
INTACT	18161	86537	8305	18896	8958	105440
MINT	8624	26698	8624	26698	62621	5661
DIP	3337	4794	1361	1468	5087	24114
BIOGRID	15239	125045	5142	11565	6248	304198

1 http://dip.doe-mbi.ucla.edu/dip/Main.cgi
2 http://www.ebi.ac.uk/intact/
3 http://mint.bio.uniroma2.it/mint/Welcome.do
4 http://ophid.utoronto.ca/ophidv2.201
5 http://thebiogrid.org

6. Applications in Biology and Medicine

6.1. Individuation of Protein Complexes

As introduced by Hartwell [22], a functional module is a group of cellular components to which a specific biological function can be attributed. Consequently, molecular interactions networks can be organized in a set of modules of a small number of participants which are low interacting with other modules.

A protein complex is a group of two or more associated proteins which interact sharing the same biological goal. For example the Breast Cancer Protein 1 (BRCA1) is known to participate in multiple cellular processes by multiple protein complexes, such as in association with the BARD1 protein or with the Rad50-Mre11-Nbs1 proteins [23].

Starting from a PPI network, complexes may be identified by searching for small and highly interconnected regions, said *cliques* [24]. Predicted complexes can be already known, i.e. their composition are known, or can denote a new protein complex. In this case, if the experiments confirm this relation, the algorithms can be used as predictors. Examples of algorithms for individuation of protein complexes are the Molecular Complex Detection algorithm (MCODE)[6], described in the early work of Bader et al, [24], and The Markov Cluster algorithm (MCL)[7] [25,26] finds clusters on a graph by simulating a stochastic flow and then analyzing its distribution.

6.2. Pathways and PPI visualization

A common and intuitive method for studying PPIs and pathways is through visualization, typically by representing a network as a "vertex-and-edge graph". Interactive networks visualization allows to discern new patterns and trends in a natural way. Visualization help to give easily answer to the researchers questions, for example which receptor is responsible for carrying a stress signal across the nuclear membrane? Which proteins need to be activated to lead to a choice between two possible differentiated cell types? Pathways and PPI visualization can produce as result an increasing in biological knowledge, since pathways are a representation of this knowledge, the pathway or PPI itself is improved as a direct result of research. Molecular Interaction Maps (MIMs) [27] is an example of graphical language can simply the understanding of complex biological systems. MIMs define a clear, accurate, and versatile graphical language to depict complex biological processes. In particular, MIMs has been used to generate new graphical hypotheses regarding the roles of specific molecules in the bioregulatory networks that control progression through the cell cycle, differentiation, and cell death.

7. Conclusion

In the modern genomic, biological networks data analysis and visualization play a central role in understanding the complex behaviour of the biological systems. Biological networks, can be represented as graph where proteins or molecules are represented as circle called nodes, interaction between two entities as line called edge. The main advantage in represent biological network as graph, is to reveal information buried in a vast set of data. Thus, visualization can illuminate surprising new connections, simplifying their interpretation. The actionable knowledge hide into the data can be increased by means of graph theory or data mining analysis methodologies, essential to produce more reliable and accurate knowledge on how molecules interacting among them allow to regulate cellular or genes functions.

[6] http://baderlab.org/Software/MCODE
[7] http://micans.org/mcl/

Acknowledgments: INdAM - GNCS Project 2017: "Efficient Algorithms and Techniques for the Organization, Management and Analysis of Biological Big Data (Algoritmi e tecniche efficienti per l'organizzazione, la gestione e l'analisi di Big Data in ambito biologico)".

Conflicts of Interest: The authors declare no conflict of interest.

References

1. Pastrello, C.; Otasek, D.; Fortney, K.; Agapito, G.; Cannataro, M.; Shirdel, E.; Jurisica, I. Visual Data Mining of Biological Networks: One Size Does Not Fit All. *PLOS Computational Biology* **2013**, *9*, 1–6.
2. Angles, R.; Gutierrez, C. Survey of graph database models. *ACM Comput. Surv.* **2008**, *40*, 1–39.
3. Deville, Y.; Gilbert, D.; van Helden, J.; Wodak, S.J. An overview of data models for the analysis of biochemical pathways. *Briefings in Bioinformatics* **2003**, *4*, 246–259.
4. Res, N.A. NAR Database ISSUE, 2007.
5. S.A..; Kauffman. Metabolic stability and epigenesis in randomly constructed genetic nets. *Journal of Theoretical Biology* **1969**, *22*, 437 – 467.
6. Tsay JJ, Wu BL, J.Y. Hierarchically organized layout for visualization of biochemical pathways. *Artif Intell Med.* **2009 Dec 29.**, *48*, 07–17.
7. Becker, M.Y.; Rojas, I. A graph layout algorithm for drawing metabolic pathways. *Bioinformatics* **2001**, *17*, 461–467, [http://bioinformatics.oxfordjournals.org/content/17/5/461.full.pdf+html].
8. Sugiyama, K.; Tagawa, S.; Toda, M. Methods for Visual Understanding of Hierarchical System Structures. *Systems, Man and Cybernetics, IEEE Transactions on* **1981**, *11*, 109 –125.
9. Fruchterman, T.M.J.; Reingold, E.M. Graph drawing by force-directed placement. *Softw. Pract. Exper.* **1991**, *21*, 1129–1164.
10. Eades, P. A Heuristic for Graph Drawing. *Congressus Numerantium* **1984**, *42*, 149–160.
11. Kamada, T.; Kawai, S. An algorithm for drawing general undirected graphs. *Inf. Process. Lett.* **1989**, *31*, 7–15.
12. Edward M. Reingold, J.S.T. Tidier Drawing of Trees. *IEEE Transaction on software engineering* **1981**, *SE-7*, 223–228.
13. Clemencon, S.; De Arazoza, H.; Rossi, F.; Tran, V.C. Hierarchical clustering for graph visualization. Proceedings of XVIIIth European Symposium on Artificial Neural Networks (ESANN 2011); , 2011; pp. 227–232.
14. Li, W.; Kurata, H. A grid layout algorithm for automatic drawing of biochemical networks. *Bioinformatics*, *21*, 2036–2042, [http://bioinformatics.oxfordjournals.org/content/21/9/2036.full.pdf+html].
15. Berg, J.; Lassig, M. Local graph alignment and motif search in biological networks. *Proc. Natl. Acad. Sci* **2004**, *41*, 14689–14694.
16. Koyutï¿rk, M.; Kim, Y.; Topkara, U.; Subramaniam, S.; Szpankowski, W.; Grama, A. Pairwise Alignment of Protein Interaction Networks. *Journal of Computational Biology* **2006**, *13*, 182 –199.
17. Agapito, G.; Guzzi, P.H.; Cannataro, M. Visualization of protein interaction networks: problems and solutions. *BMC bioinformatics* **2013**, *14*, S1.
18. Pavlopoulos, G.A.; Secrier, M.; Moschopoulos, C.N.; Soldatos, T.G.; Kossida, S.; Aerts, J.; Schneider, R.; Bagos, P.G. Using graph theory to analyze biological networks. *BioData Mining* **2011**, *4*, 10.
19. Guzzi, P.H.; Milenković, T. Survey of local and global biological network alignment: the need to reconcile the two sides of the same coin. *Briefings in Bioinformatics* **2017**, p. bbw132.
20. Mina, M.; Guzzi, P.H. Improving the robustness of local network alignment: design and extensive assessment of a markov clustering-based approach. *IEEE/ACM Transactions on Computational Biology and Bioinformatics (TCBB)* **2014**, *11*, 561–572.
21. Cho, Y.R.; Mina, M.; Lu, Y.; Kwon, N.; Guzzi, P.H. M-finder: Uncovering functionally associated proteins from interactome data integrated with go annotations. *Proteome science* **2013**, *11*, S3.
22. Hartwell, L.; Hopfield, J.; Leibler, S.; Murray, A. From molecular to modular cell biology. *Nature* **1999**, *402*, C47–C52.
23. Cortez, D.; Wang, Y.; Qin, J.; Elledge, S. Requirement of ATM-dependent phosphorylation of brca1 in the DNA damage response to double-strand breaks. *Science* **1999**, *286*, 1162–6.

24. Bader, G.; Hogue, C. An automated method for finding molecular complexes in large protein interaction networks. *BMC Bioinformatics* **2003**, *4*, 2.
25. Enright, A.J. Van Dongen, S.; Ouzounis, C. An efficient algorithm for large-scale detection of protein families. *Nucleic Acids Research* **2002**, *30*, 1575–1584.
26. van Dongen, S. Graph Clustering by Flow Simulation. PhD thesis, , May 2000. PhD thesis, University of Utrecht, May 2000.
27. Aladjem, M.I.; Pasa, S.; Parodi, S.; Weinstein, J.N.; Pommier, Y.; Kohn, K.W. Molecular interaction maps–a diagrammatic graphical language for bioregulatory networks. *Sci. STKE* **2004**, *2004*, pe8–pe8.