A Guide to Conquer the Biological Network Era Using Graph Theory

Mikaela Koutrouli 1†, Evangelos Karatzas 1,2†, David Paez-Espino 3 and Georgios A. Pavlopoulos 1*

1 Institute for Fundamental Biomedical Research, BSRC “Alexander Fleming”, Vari, Greece, 2 Department of Informatics and Telecommunications, University of Athens, Athens, Greece, 3 Lawrence Berkeley National Laboratory, Department of Energy, Joint Genome Institute, Walnut Creek, CA, United States

Networks are one of the most common ways to represent biological systems as complex sets of binary interactions or relations between different bioentities. In this article, we discuss the basic graph theory concepts and the various graph types, as well as the available data structures for storing and reading graphs. In addition, we describe several network properties and we highlight some of the widely used network topological features. We briefly mention the network patterns, motifs and models, and we further comment on the types of biological and biomedical networks along with their corresponding computer- and human-readable file formats. Finally, we discuss a variety of algorithms and metrics for network analyses regarding graph drawing, clustering, visualization, link prediction, perturbation, and network alignment as well as the current state-of-the-art tools. We expect this review to reach a very broad spectrum of readers varying from experts to beginners while encouraging them to enhance the field further.

Keywords: biological networks, topology, graph theory, visualization, clustering

INTRODUCTION

While most recent review articles focus on biomedical and biological networks and their applications (McGillivray et al., 2018; Sonawane et al., 2019; Yue et al., 2019), in certain case studies, familiarity with the graph theory concepts behind these networks is often missing. The aim of this review is to tackle questions raised by today’s increasing demands and aid researchers in understanding the graph theory behind the biomedical networks as well as concepts such as visualization, annotation, management, clustering, integration, etc. To do this, we start with an introduction about graphs (in discrete mathematics) and their different types and we further describe the various data structures and file formats for storage and representation. In addition, we discuss several topological features and network properties, as well as concepts such as graph clustering, clustering comparison, network alignment, motif detection, and edge prediction. We further comment on the various layout and graph drawing techniques as well as on methods regarding network alignment and link predictions and we highlight the state-of-the-art tools for analyzing such networks. Finally, we try to bring graph theory into a biomedical context by providing a thorough description about the different types of biomedical networks and the sources used for their construction. We hope this review becomes a useful handbook for readers regardless of their scientific background and help non-experts in handling and interpreting networks more easily.

For example, in general, networks or graphs (mathematical way of representing a network) are used to capture relationships between entities or objects. In a typical representation, a graph is composed of a
set of vertices/nodes/points, connected with edges/lines/links/arrows/arts. Examples of networks which we interact with in everyday life include the electricity grid, road maps, the world wide web, the internet, airline connections, citation and language networks, telecommunication channels, social networks, economic networks, and many others. Graph theory has been the established mathematical field for the study and the analysis of such networks and is applicable to a wide variety of disciplines, ranging from mathematics, physics, computer science, engineering, and sociology to biology and medicine (Junker and Schreiber, 2008; Pavlopoulos et al., 2011a). In the biomedical field for example, many biological networks consist of molecules such as DNA, RNA, proteins and metabolites, and graphs can be used to capture the interactions between these molecules. Therefore, it is essential to know the various network types which can be used, in order to be able to communicate and visualize such interactions.

Starting with the basic notions, in mathematics, a set \( A = \{a_1, a_2, a_3, \ldots a_n\} \) is a collection of objects \( a_1, a_2, a_3, \ldots a_n \), whereas a graph \( G = (V, E) \) is composed of a set of vertices \( V \) and a set of edges \( E \). A subgraph \( G' = (V', E') \) of the graph \( G = (V, E) \) is a graph where \( V' \) is a subset of \( V \) and \( E' \) a subset of \( E \). While one graph can have multiple representations, two different graphs may be isomorphic if they contain the same number of vertices connected in the same way. Examples are shown in Figures 1A–C.

There are various graph categories. The most known are undirected, directed, weighted, bipartite, multi-edge, hypergraphs, and trees.
A graph is **undirected** if there is a single connection defined as \( E = \{(i, j) | i, j \in V\} \) between vertices \( i \) and \( j \). In such case, vertices \( i \) and \( j \) are called direct neighbors (e.g., gene co-expression network).

A graph is called **directed** if an edge between vertices \( i \) and \( j \) is represented by an arrow, thus indicating a direction from vertex \( i \) to vertex \( j \) or vice versa. A directed graph is defined as an ordered triple \( G = (V, E, f) \) where \( f \) is a function that maps each element in set \( E \) to an ordered pair of vertices in \( V \) (e.g., pathway).

Notably, in biology there are a number of directed relationships which can be graphically shown as different arrow types toward a semantic approach (e.g., food web). For example, “inhibits,” “enhances,” “regulates” etc. Standards for arrow usage are described in the Systems Biology Graphical Notation (SBGN) visual language (Le Novère et al., 2009).

A weighted graph is defined as a graph where \( E \) is a set of edges between the vertices \( i \) and \( j \) (\( E = \{(i, j) | i, j \in V\} \)) associated with a weight function \( w: E \rightarrow \mathbb{R} \), where \( \mathbb{R} \) denotes the set of all real numbers. Most of the times, the weight \( w_{ij} \) of the edge between nodes \( i \) and \( j \) represents the relevance of the connection (e.g., sequence similarity network).

A bipartite graph is an undirected graph \( G = (V, E) \) in which vertices in \( V \) can be partitioned into two sets \( V' \) and \( V'' \) such that \( (i, j) \in E \) implies either \( i \in V' \) and \( j \in V'' \) or \( (i \in V' \) and \( i \in V'' \) (e.g., gene-disease networks). In other words, any vertex from set \( V' \) can be connected to any other vertex from set \( V'' \) but no edges between vertices within the same set (\( V' \) or \( V'' \)) are allowed.

A graph is called **multi-edge** if it contains multiple edges or otherwise parallel edges that are incident to the same two vertices (e.g., knowledge/integration networks). A simple graph for example, has no multiple edges.

A hypergraph consists of a set of vertices \( V \) and a set of hyperedges \( E \) where an edge can join any number of vertices (e.g., biochemical networks).

A **tree** is an undirected graph in which any two vertices are connected by exactly one path, or equivalently a connected acyclic undirected graph (e.g., ontologies, phylogenies). Examples of the various graph types are shown in Figures 2D–L.

A graph is **connected** if there is a path from any point to any other point in the graph. In a **complete graph**, every pair of distinct vertices is connected by a unique edge.

A **cluster** (Figure 1M) is a graph formed from the disjoint union of complete graphs and a **clique** (Figure 1N) in an undirected graph is a subset of vertices such that every pair of vertices in the clique is connected.

### DATA STRUCTURES AND REPRESENTATIONS

A network can be stored as (i) adjacency matrix, (ii) adjacency list, or (iii) sparse matrix. In graph theory, an adjacency matrix \( A \) is a square matrix of size \( N \times N \) (where \( N \) is the number of vertices) used to represent a graph. In the case of a simple graph, the adjacency matrix is a \((0, 1)-\)matrix with zeros on its diagonal \( A[i, j] = 1 \) for connection presence, \( A[i, j] = 0 \) for connection absence) or a \((0, w_{ij})\)-matrix for a weighted graph where \( w_{ij} \) is the edge weight between two nodes \( A[i, j] = w_{ij} \). In both undirected simple and weighted graphs, the adjacency matrix is symmetric (equal to its transpose-rows and columns are the same). In the case of directed graphs, the matrix is not symmetric, thus differentiating its upper triangular part from its lower triangular part \( (ij \text{ is not the same as } ji) \).

An overview of adjacency matrices and their representations are shown in Figures 2A–C.

Bipartite graphs, as opposed to generic networks, have their own characteristics (Pavlopoulos et al., 2018). One major property is that any bipartite graph can be presented as two biadjacency matrices (or otherwise projections). While in an original bipartite graph, vertices which belong to a set are not connected to each other, in its biadjacency form they are connected through nodes that belong to the other set (indirect connections). This concept is described in Figures 2D,E, whereas an extensive review about their biomedical application can be found elsewhere (Pavlopoulos et al., 2018).

Adjacency matrices are memory inefficient for storing larger sparse networks as they require \( O(V^2) \) memory. Notably, the adjacency notation in graph theory is a theoretical measure to classify algorithms according to how their running time or space requirements grow as the input size grows (Knuth, 1997). Let’s assume that in a gene co-expression network, one wants to store an all-vs.-all matrix with all pairwise human gene similarities \( (V = \sim 20,000 \text{ genes}) \). This would require 400,000,000 bytes to be stored in memory (381 MB RAM) or \( 4 \times 0.000918 \text{ (GB)} = 1.49 \text{ GB} \) for float/integer numbers (for 4 byte integers and floats). To partially overcome this barrier, a simple approach would be to take advantage of the adjacency matrix symmetry by only storing the upper triangular part in an array \( B \) in a linear form (Figures 2F,G). The mapping between element coordinates in the two forms is given by the formula \( A[i, j] = B[Ni + \frac{(1-i)}{2} + (j - i)] \) where \( N \) is the number of vertices (Figure 2G).

The linear representation \( B \) requires \( \frac{V(V-1)}{2} \) memory which is half the size compared to the memory needed for a complete adjacency matrix \( A \).

For sparse networks, adjacency lists are proposed as an alternative data structure. An adjacency list is an array \( A \) of separate lists. Each element of the array \( A_i \) is a list, which contains all the vertices that are adjacent to vertex \( i \). If the graph \( G \) is weighted, then each item in the adjacency list is either a two-item array or an object, giving the vertex number, and the edge weight (Figure 2H). Adjacency lists require much less space \( O(V+E) \) compared to the space required by the adjacency matrix \( O(V^2) \).

Moreover, finding all vertices adjacent to a given vertex in an adjacency matrix representation, requires \( O(V) \) time, whereas in an adjacency list such operation is as fast as reading the corresponding list (smaller length).

An alternative to the adjacency list, is the use of a sparse matrix data structure. In such case only the non-zero elements are kept along with their coordinates and everything else is discarded as non-informative. An example of such a data structure is shown in Figure 2I where the first row keeps the \( i \) coordinate for each element in \( A[i, j] \), the second row the
As degree $d_{ki}$, we define the total number of edges adjacent to a vertex. In the case of a directed graph we distinguish between the “indegree” ($d_{ki}$) and “outdegree” ($d_{ko}$). The indegree refers to the number of arcs, incident from the vertex, whereas the outdegree to the number of arcs incident to the vertex. In a social network for example, the indegree would represent the followers, whereas the outdegree the people one follows. The total degree in a directed graph is the sum of the indegree and outdegree $d_k = d_{ki} + d_{ko}$ showing all connections (both followers and followed people). The average degree of the network is $d_{avg} = \frac{\sum d_k}{n}$ (Figure 3A).

Looking at all nodes in a network, in order to study the degree distribution $p(k)$, we consider the probability that a randomly selected vertex has degree equal to $k$. The same information can also be found as cumulative degree distribution $p_c(k)$ which shows the a-posterior probability of a randomly selected vertex to have degree larger than $k$. Notably, the degree distribution is one of the most important topological features and is characteristic to different network types. In the simplest case, $p(k)$ can be estimated by a histogram of degrees. An example is shown in Figure 3B. Networks, whose degree distribution follow a power law, are called scale-free networks.
Density is the ratio between the number of edges in a graph and the number of possible edges in the same graph. In a fully connected graph (e.g., protein complex), the number of possible edges (pairwise connections) are $E_{\text{max}} = \frac{V(V-1)}{2}$. Therefore, the density can be calculated as $\text{density} = \frac{E}{E_{\text{max}}} = \frac{2E}{V(V-1)}$. If a graph has $E \simeq V^k$, $2 > k > 1$, then this graph is considered as dense, whereas when a graph has $E \simeq V$ or $E \simeq V^k$, $k \leq 1$, it is considered as sparse.

The Clustering coefficient is a measure which shows whether a network or a node has the tendency to form clusters or tightly connected communities (e.g., protein clusters in a protein-protein interaction network). The clustering coefficient of a node is defined as the number of edges between its neighbors divided by the number of possible connections between these neighbors. The clustering coefficient of a node $i$ is defined as $C_i = \frac{k_i(k_i-1)}{2e}$, where $k$ is the number of neighbors (degree) and $e$ the number of edges between these $k$ neighbors. The average clustering of a network is given by $C_{\text{avg}} = \frac{\sum_i C_i}{N}$. The clustering coefficient takes values $0 \leq C_i \leq 1$, thus the closer to 1, the higher the tendency for clusters to be formed. An example is shown in Figures 3C,D.

The matching index $M_{ij}$ can be used to identify two nodes in a network which might be functionally similar without necessarily being connected to each other. The matching index is a measure to quantify such similarity between any two nodes within a network and, according to the above, two nodes can be found to be functionally similar if they share common neighbors. The matching index between vertices $i$ and $j$ is calculated as $M_{ij} = \frac{\Sigma \text{ distinct common neighbors}}{\Sigma \text{ total number of neighbors}}$ and can be extended beyond the direct neighbors of a vertex. In addition, it can be applied to multi-edge networks.

The distance $\text{dist}_{ij}$ between two nodes (e.g., metabolites in a metabolic network) is defined as the length of the shortest path between them. As shortest path we define the minimal number of edges that need to be traversed to reach node $j$ from node $i$. In the case where two shortest paths of identical length exist, any of them could be used. Whenever there is no connection between two nodes $i$ and $j$, then their distance is defined as infinite $\text{dist}_{ij} = \infty$. In addition, the diameter, $\text{diam}_{m} = \max(\text{dist}_{ij})$, is the maximal distance between any pair of vertices. The average path length is defined as the average distance between all node pairs and is defined as $\text{dist}_{\text{avg}} = \frac{1}{N(N-1)} \sum_{i=1}^{N} \sum_{j=1}^{N} \text{dist}_{ij}$. 

*Frontiers in Bioengineering and Biotechnology | www.frontiersin.org*
NETWORK CENTRALITIES

Very often, in network analysis, we ask questions such as: which is the most important node, which node behaves as a hub, which node is the bridge between two different communities, which node is important for the network's robustness (tolerance to failures and perturbations), etc. In order to address these questions, various network centralities can be used. The degree centrality (Bonacich, 1987) is a measure to highlight highly connected nodes (e.g., central transcription factors). A network with a star-like topology for example, contains hubs which are central nodes with many neighbors around them. The degree centrality of a node $i$ is calculated as $C_i = \text{deg}(i)$ where $\text{deg}(i)$ is the node's degree. Similarly, the closeness centrality (Sabidussi, 1966) is a measure to detect important nodes which can communicate quickly with other nodes in a network. For a graph $G = (V,E)$ it is defined as $C_{\text{clo}} = \frac{1}{\sum_{dist_{ij}}}$ or as $C_{\text{clo}} = \frac{N-1}{\sum_{dist_{ij}}}$ in its normalized form. In biochemical networks, it is often used to find top metabolites [e.g., metabolites in E. coli as part of the glycolysis and citrate acid cycle pathways (Ma and Zeng, 2003; Koschützki and Schreiber, 2008)]. The betweenness centrality (Freeman, 1977) shows the nodes which form such bridges so that two communities can communicate with each other. It is calculated as $C_{\text{bet}}(i) = \frac{\sigma_{xy}(i)}{\sigma_{xy}}$ where $\sigma_{xy}$ is the total number of shortest paths from node $x$ to node $y$ and $\sigma_{xy}(i)$ is the number of those paths that pass through node $i$. It has been shown that proteins with high betweenness centrality in a protein-protein interaction (PPI) network play an important role to the modularization of the network (Koschützki and Schreiber, 2008). The eccentricity centrality (Hage and Harary, 1995) shows how easily accessible a vertex is from any other vertex in the network. The eccentricity is the maximum graph distance between vertex $i$ and any other vertex $j$ in graph $G$. For a disconnected graph, all vertices are defined to have infinite eccentricity. The eccentricity centrality is calculated as $C_{\text{ecc}} = \frac{1}{\max(dist_{ij})}$. Eccentricity centrality has been used to detect essential proteins in a PPI (Jalili et al., 2016). Notably, the maximum eccentricity is called the graph diameter, whereas the minimum graph eccentricity is called the graph radius. Finally, there are many other specialized centralities that serve different purposes. The eigenvector centrality for example, detects vertices that are connected to important vertices, whereas the subgraph centrality accounts for the participation of a node in all subgraphs of the network. Examples are shown in Figure 3E.

FIGURE 4 | Motifs. (A) Motif examples of three and four nodes. (B) The 13 possible directed motifs using three nodes.

MOTIFS

Network motifs are repeated graphlets (small subgraphs of a larger network that appear at any frequency) in a specific network capturing particular patterns of interactions between vertices. They are often associated with particular functions (Stone et al., 2019) and are used for many applications in biological networks (Kim et al., 2011). Motifs are structures which occur at higher frequencies compared to random networks and are found in both directed and undirected networks. Motif analysis is often applied on biological networks such as biochemical, ecological, neurobiology, or gene expression networks to unravel building blocks associated with certain biological processes.

For example, motifs can be found in ecological food webs as well as genetic networks or the World Wide Web (Milo, 2002; Shen-Orr et al., 2002). Feed-forward-loop (FFL) and bifan motifs (Figure 4) are typical patterns found in various types of biological networks (Mangan and Alon, 2003; Mangan et al., 2003). Notably, motifs have been used to distinguish different protein-protein interaction networks (Przulj et al., 2004) and in contrast to the transcriptional regulatory networks, it has been shown that they are evolutionary conserved in PPI networks (Conant and Wagner, 2003).

To measure the statistical significance of a network motif, a Z-score or a $P$-value can be used. The Z-score is calculated as the difference of the frequency $f(m)$ of a motif $m$ in a network and its mean frequency $f_e(m)$ in a large number of
randomized networks \( \sigma_r(m) \). The formula is \( Z(m) = \frac{f(m) - f_r(m)}{\sigma_r(m)} \). Similarly, the \( P \)-value shows the probability \( P(m) \) of a motif \( m \) to appear in a randomized network equally or more times than in the network of interest. Motifs are considered to be statistically significant if they have \( Z(m) > 2.0 \). Motif detection can become computationally expensive and tools like Pajek (Mrvar and Batagelj, 2016), Mfinder (Jeong et al., 2000), and NetMatch (Schreiber and Schwöbbermeyer, 2005) have been introduced for this purpose. And tools like Pajek (Mrvar and Batagelj, 2016), Mfinder (Jeong et al., 2000), and NetMatch (Schreiber and Schwöbbermeyer, 2005) have been introduced for this purpose. In a Watts-Strogatz network, two vertices are randomly connected with probability \( p \) and show that the probability of two nodes with a common neighbor to be connected is the same as the probability of two randomly paired vertices. In the case of biological networks, straightforward comparisons show if they have a certain topology or differ from any other random network. Thus, Erdos–Rényi is not a good model for biological networks with respect to degree distribution.

The Watts–Strogatz model: This model was introduced to describe random networks that follow a small world topology meaning that most nodes can be reached by any other node in a small number of steps. While random networks can often capture this property too, they fail to account for highly connected regions like in most empirical networks (e.g., social networks). Therefore, Watts and Strogatz proposed a model for networks described by local structures (high clustering coefficient) as well as small average path lengths. Metabolic networks [e.g., fat-metabolism communication in Yeast (Al-Anzi et al., 2015)], in which metabolites are linked to each other with small steps, is a typical example (Jeong et al., 2000). In a Watts-Strogatz network, if all vertices are placed on a circular ring, each vertex would be connected to its \( \frac{V}{2} \) neighbors. In the real world, this indicates the form of small communities where people know other people from their close environment as well as friends of friends from nearby areas. Coexistence of high local clustering and short average path length are two main characteristics of this type of networks.

The Barabási–Albert model: This model describes random scale-free networks. These are networks whose degree distribution follows a power law taking into account their inhomogeneous degree distribution or otherwise networks with nodes which do not have a typical number of neighbors. According to this model, networks can evolve overtime and new edges do not appear randomly, whereas new nodes follow the existing degree distribution. At time point \( t = 0 \) for example, let’s assume a network consisting of \( V_0 \) vertices and zero edges. A new vertex will connect with \( e \leq V_0 \) edges to the existing vertices, whereas after \( t \) time points, the network is expected to consist of \( V = V_0 + et \) edges.
Notably, for $t \gg 1$, the Barabasi-Albert model will exhibit a scale-free distribution $p(k) \sim k^{-\gamma}$, $\gamma = 3$. Like in a social network, individuals who already have many friends are likely to acquire more friends overtime compared to individuals with a limited number of friends. When comparing the Erdos–Renyi and Watts-Strogatz networks of the same size and density, the Barabasi-Albert networks were found to have shorter average path lengths. Characteristic examples of BA networks are the Protein-Protein interaction networks (Barabasi and Oltvai, 2004; Yook et al., 2004).

Like in many real-life examples, most biological networks are robust and tolerant against random removal of nodes as biological functions must remain maintained. However, compared to random networks with homogeneous degree distribution, scale-free networks are very vulnerable to targeted attacks but very robust against random removal of vertices. In general, nodes with low degree appear more frequently compared to nodes with high degree and play a minor role in the overall network topology, whereas aimed removal of nodes with higher degree distribution can affect a network’s topology significantly.

**BIOLOGICAL AND BIOMEDICAL NETWORKS**

In biomedical research, graphs can capture the associations between any type of biological entity such as proteins, genes, small molecules, metabolites, ligands, diseases, drugs, or even database records (Figure 6).
Some biological networks model the functions of cellular and tissue-specific molecular interactions at a cellular organizational level, varying from cells to a complete organ. These are:

### Protein-Protein Interaction Networks (PPIs)
This type of networks holds information about how different proteins operate with each other to enable a biological process within a cell. The interactions in a PPI network can be physical or predicted. Notably, a whole interactome can capture all PPIs happening in a cell or an organism. In vivo and in vitro methods for detecting PPIs include: X-ray crystallography, NMR, tandem affinity purification (TAP), affinity chromatography, co-immunoprecipitation, protein arrays, protein fragment complementation, phage display and yeast two-hybrid (Y2H) (Rao et al., 2014). Widely used repositories (Lehne and Schlitt, 2009; Szklarczyk and Jensen, 2015) which host PPIs for various organisms are the BioGRID (Stark et al., 2006), MINT (Chatr-aryamontri et al., 2007), BIND (Bader et al., 2003), DIP (Xenarios et al., 2000), IntAct (Hermjakob et al., 2004a), and HPRD (Peri et al., 2003) database. Concerning topology, the PPI networks follow a small-world property and are scale-free networks. Central hubs often represent evolutionarily conserved proteins, whereas cliques (fully connected subgraphs) have been found to have a high functional significance (Spirin and Mirny, 2003).

### Sequence Similarity Networks (SSNs)
These networks consist of nodes representing proteins or genes and edges capturing the sequence similarity between amino acid or nucleotide sequences. Widely used tools (Ekre and Mante, 2016) for obtaining a sequence similarity between two sequences are the BLAST (Altschul et al., 1990), LAST (Kieferbasa et al., 2011), and FASTA3 suite (Pearson, 2000), which contains SSEARCH, GGSEARCH, GLSEARCH executables of Smith-Waterman (Smith and Waterman, 1981) and Needleman-Wunsch (Needleman and Wunsch, 1970) implementations for local and global sequence alignment. These networks are weighted, have a small-world and scale-free topology and often contain hubs. Often, clustering algorithms are applied on such networks for the detection of protein families. Like in PPIs, proteins that lie together in such networks are more likely to have similar functions or be involved in similar biological processes (Sharan et al., 2007). While it is not straightforward to come to a conclusion about their density, when coping with fragmented sequences (e.g., alignments of predicted proteins from metagenomes), the networks are rather sparse.

### Gene Regulatory Networks
They are collections of regulatory relationships between transcription factors (TFs) and TF-binding sites or between genes and their regulators. Normally, these networks are directed, dynamic, and can be visualized as bipartite graphs. In such networks, most nodes have only a few interactions and only a few hubs come with a higher connectivity degree. In any case, such networks follow a power law degree distribution (scale-free) \( p(k) \sim k^{-\gamma}, \gamma \approx 2 \) (Vázquez et al., 2004). Among a variety of databases hosting information about gene regulation, widely used repositories are the KEGG (Kanehisa and Goto, 2000), GTRD (Yevshin et al., 2019), TRANSFAC (Matys et al., 2003), TRRUST (Han et al., 2018).

### Signal Transduction Networks
These networks capture cell signaling or otherwise the transmission of molecular signals as well as a series of molecular events within a cell or from the exterior to its interior (Fabregat et al., 2018). A signal transduction network normally consists of several thousand nodes and edges representing a series of reactions. These networks are mostly directed and sparse. They follow a power law degree distribution as well as small-world properties. While such data can be found in well-known pathway databases (KEGG, Reactome), specialized repositories such as the MIST (signal transduction in microbes) (Ulrich and Zhulin, 2007), NetPath (Kandasamy et al., 2010), or Human-gpDB (Satagopam et al., 2010) also exist.

### Metabolic Networks
They are networks consisting of metabolites (nodes) and their interactions in an organism. Metabolites can be either smaller molecules such as amino acids or larger macromolecules like polysaccharides. These networks are usually directed graphs and can be represented as Petri nets (Reisig, 1985; Chauviya, 2007). They are scale-free, they carry small-world properties (Jeong et al., 2000) and can often be organized using hierarchies (Gagneur et al., 2003). In order to gain insights into their decomposition, heuristic modularity optimization over all possible divisions to find the best one is required (Newman and Girvan, 2004). KEGG and Reactome databases are two of the most widely used repositories for this type of network.

### Gene Co-expression Networks
They are undirected weighted networks where two nodes (genes) are connected if there is a significant co-expression between them. Such networks are usually constructed using data from high-throughput technologies such as Microarrays, RNA-Seq or scRNA-seq. For each pairwise connection, a metric like for example, the Pearson Correlation Coefficient (PCC) (Kirch, 2008) can be used to calculate an edge’s weight. Often, a threshold or a Z-score are applied on the whole network in order to accept correlations above a certain cutoff. Otherwise the network would look like a fully connected clique. After the threshold and depending on the total clustering coefficient, the network can be clustered to detect functional modules. One typical example is the ribosomal genes which tend to group together due to similar expression patterns. Expression data for such analyses can be found in widely used repositories such as GEO (Barrett et al., 2013) or ArrayExpress (Parkinson et al., 2007). Notably, Arena-Idb (Bonnici et al., 2018) repository can be used for human non-coding RNAs interactions.

### Expression Quantitative Trait Loci (eQTL) Network
Data obtained from genotyping and/or transcriptomic experiments are used as locus (eQTLs) in explaining a fraction
of the genetic variance of a gene expression phenotype (Nica and Dermitzakis, 2013). For this purpose, eQTL networks are suitable for summarizing this information (Platig et al., 2016; Fagny et al., 2017; Sonawane et al., 2019). Genome-wide association studies (GWASs) are used for association between common genetic variants and phenotypic traits based on many variants of relatively small effect size. Those single-nucleotide polymorphisms (SNPs) are measured by expression quantitative trait locus (eQTL) analysis and are represented by eQTL networks with significant associations as edges. Findings provide unique insight into the genotype–phenotype relationship [e.g., Enhanced tissue-specific heritability of type 2 diabetes (T2D) was identified by eQTL networks (Torres et al., 2014; Fagny et al., 2017)].

**IncRNA–Protein Interaction Networks**

These networks reveal the functions of IncRNAs coming from their interactions with proteins (Yue et al., 2019). Most common experiments for studying these interactions include the RNP immunoprecipitation-microarray (RIP-Chip) (RIP-Chip: the isolation and identification of mRNAs, microRNAs, and protein components of ribonucleoprotein complexes from cell extracts), high-throughput sequencing of RNA isolated by crosslinking immunoprecipitation (HITS-CLIP) (HITS-CLIP yields genome-wide insights into brain alternative RNA processing.), photoactivatable ribonucleoside-enhanced crosslinking and immunoprecipitation (PAR-CLIP) (Transcriptome-wide identification of RNA-binding protein and microRNA target sites by PAR-CLIP.) and RNAcompete (Rapid and systematic analysis of the RNA recognition specificities of RNA-binding proteins). Regarding the computational methods for predicting these interactions, network-based methods are the most applicable. *Multiple protein–protein similarity networks (PPSNs)* (Fusing multiple protein-protein similarity networks to effectively predict IncRNA-protein interactions.), *LPiHN* (Predicting Long Noncoding RNA and Protein Interactions Using Heterogeneous Network Model) and *PLPiHS* (Prediction of IncRNA-protein interactions using HeteSim scores based on heterogeneous networks) could be applied to generate IncRNA–protein interaction networks (Zhang et al., 2019).

Additionally, some biological networks are distinguished by comprising information about evolution and interactions of species. These are:

**Phylogenetic networks**: They are networks trying to capture the evolutionary relationships between organisms in time (Huson et al., 2010; Thomas and Portier, 2013). Reconstructed phylogenies are mainly represented as trees even if it is debatable whether a tree is the right scheme as it fails at capturing events like the union of different lineages. As an extension to trees, phylogenetic networks might contain loops. The tree of life is a global effort to capture the evolution of all organisms in a single snapshot and describe the relationships between them. Notably, widely used methods for tree reconstruction are the Neighbor-Joining (NJ) (Saitou and Nei, 1987), UPGMA, and maximum likelihood parsimony (Golding and Felsenstein, 1990), whereas widely used applications for such analyses are the PAUP (Yang, 1996), PHYLIP (Baum, 1989), and MEGA (Kumar et al., 2016).

**Ecological networks**: These networks mainly represent food webs or interactions among species in an ecosystem. These interactions can be trophic or symbiotic (Ings et al., 2009), mutualistic (bidirectional) or competitive (host-parasite). A fundamental aim of ecological network analysis is to uncover the mechanisms which influence the stability of fragile ecosystems. In general, binary food webs can be simple directed or undirected k-partite or simple graphs, whereas quantitative food chains can be shown as weighted graphs. Most food webs follow an exponential degree distribution, whereas it is well-accepted that such webs display an average low connectance. An in-depth analysis of the topological features of this type of networks is extensively discussed elsewhere (Danon et al., 2011).

**Epidemiological networks**: They are networks used in public health to study disease transmission (e.g., sexually transmitted diseases—STDs) (Danon et al., 2011). Path traversal analysis can reveal transmission routes while the network’s structure can provide insights into the epidemiological dynamics. While epidemiological networks often simulate social networks, they can be shown as bipartite graphs (Pavlopoulos et al., 2018).

**Species interaction networks**: There are *between-species interaction networks* describing pairwise interactions between species, trying to understand what factors (e.g., diversity) lead to stability (Romanuik et al., 2010) and *within-species interaction networks* quantifying associations between individuals, offering information in species, and/or population level (Croft et al., 2004).

**Food webs**: All organisms are connected to each other through feeding interactions and the networks presenting these interactions are very—well known for the effort to answer the long-standing question in ecology about the stability of these interactions (Milner-Gulland, 2012). Interactions of ecological entities captured in networks can be obtained from literature articles, observation in the field, molecular experiments (e.g., analysis of environmental DNA), or models based on incomplete data (Delmas et al., 2019).

Moreover, biomedical graphs are of great importance for both researchers and clinicians (Yue et al., 2019). These are:

**Disease networks**: They are formed by diseases and their causative genes, while the connections between them can be constructed based on repositories such as the Online Mendelian Inheritance in Man (OMIM) associations. These networks are generated when diseases share at least one causative gene, and therefore are considered to be linked. Disease networks are typically shown as bipartite networks (Goh et al., 2007).
Drug-disease associations: These networks hold information about known and/or predicted drug-disease associations. The information could be extracted from a database or from published literature (Gottlieb et al., 2011; Sonawane et al., 2019; Yue et al., 2019).

Disease—symptom graphs: These graphs connect diseases with their symptoms and visualize the potential evolution of the diseases, assisting clinicians to follow the more efficient medical treatment rapidly (Sonawane et al., 2019). These graphs are generated based on medical records using rudimentary concept extraction of cause and effect.

Finally, data integration approaches can be used to generate biological networks consisting of nodes which represent text, database records, or literature articles.

Literature co-occurrence networks: These networks show connections between bioentities that are found to co-occur in any text corpus (Pavlopoulos et al., 2014). Name-entity-recognition (NER) tags such as the EXTRACT (Pafilis et al., 2016), can be used to initially identify genes/proteins, chemical compounds, environments, tissues, diseases, phenotypes, and Gene Ontology terms in a text and map the identified terms to their corresponding ontology/taxonomy entries in public databases. This way, any text corpus like Wikipedia, PubMed (~29 million abstracts) or PubMed Central (PMC, ~6 million full-text articles) can be parsed and analyzed for both abstract-based or sentence-based co-occurrences.

Knowledge networks: These networks are mostly multi-edge graphs as they combine heterogeneous information and metadata from various sources like public repositories or biological and literature databases. Typical examples are the STRING (Franceschini et al., 2013), STITCH (Szklarczyk et al., 2016), and PICKLE (Gioutlakis et al., 2017) databases. STRING contains known and predicted protein-protein interactions for various organisms, whereas STITCH contains known and predicted interactions between chemical compounds and proteins. In the STRING database, two proteins can be, for example, connected in multiple ways. They can be homologous, or co-occur in an abstract, or have neighboring positions in a genome or be products of a fusion event or co-express in an experiment. Similarly, PICKLE integrates publicly available PPI databases via genetic information ontology. Finally, bioDBnet (Mudunuri et al., 2009) is a network of the major biological databases.

Overall, biological networks follow the new era of hybrid heterogeneous networks, trying to put together different types of information (Navlakha and Kingsford, 2010; Moreau and Tranchevent, 2012; Ni et al., 2016). It is worth mentioning, that a great collection of biological networks that are produced by researchers and are published in various articles can be found in https://cytoscape-publications.tumblr.com. This repository can be used as an excellent teaching material as well as a great resource for inspiration and case studies when building software applications.

FUNCTIONAL ANNOTATION AND OVERREPRESENTATION ANALYSIS

A common task in computational biology field is the annotation and interpretation of gene lists (e.g., genes or proteins which are found to be tightly connected in a network). For this task, functional annotation and/or overrepresentation analysis can be used (Tipney and Hunter, 2010; Hung et al., 2012). Enrichment analysis determines over-represented classes of genes or proteins in a large group of samples in order to reveal existing associations with disease phenotypes (Huang et al., 2009a). Similarly, functional enrichment analysis applies statistical tests to match genes of interest with certain biological functions (Bindea et al., 2009). PANTHER (Mi et al., 2013), Gorilla (Eden et al., 2009) and DAVID (Huang et al., 2009b) applications for example, accept a gene list as an input and report related hits to molecular functions, biological processes [e.g., Gene Ontology (Gene Ontology Consortium, 2004)] and KEGG (Kanehisa and Goto, 2000) and Reactome (Fabregat et al., 2018) pathways. Another similar tool is the ClueGO (Bindea et al., 2009) which is offered as a Cytoscape plugin. For researchers interested in non-coding RNA annotation and identification, Transfector (Tripathi et al., 2015) can be used. Pathway enrichment analysis can be also performed by additional tools such as pathfindR (Ulegen et al., 2019), g:Profiler (Raudvere et al., 2019), and EnrichmentMap (Merico et al., 2010; Reinmand et al., 2019). Gene Set Enrichment Analysis [GSEA (Mootha et al., 2003; Subramanian et al., 2005)] and NGSEA (Han et al., 2019) can be used for overrepresentation analysis, whereas differential expression analysis for the determination of the up- and down- regulated genes is offered by DESeq2 (Michael, 2017) or metaseqR (Moulos and Hatzis, 2015).

FILE FORMATS

A network can be described and stored in multiple human- and computer-readable ways. Apart from the simple file formats such as the tab-delimited, CSV, SIF, Excel and adjacency matrix, several others like the BioPAX (Demir et al., 2010), SBML (Hucka et al., 2003), PSI-MI (Hermjakob et al., 2004b), CML (Murray-Rust et al., 2001), and CellML (Lloyd et al., 2004) have been introduced for biological data and semantics. For example, SBML, which stands for Systems Biology Markup Language, is an XML-like format for storing and parsing biochemical networks as well as for describing biological processes. BioPAX stands for Biological Pathway Exchange and is made for the representation of biological pathways at the molecular and cellular level. The PSI-MI format is used for the data exchange related to molecular interactions and CellML is used for describing mathematical models. GraphML (Brandes et al., 2017) is an XML-like file format and consists of unordered sections related to a network’s node and edge elements. Each node has a distinct identifier, whereas each edge is described by a source and a target node. Additional attributes such, an edge weight or a label can also be included in the schema. The JavaScript Object Notation (JSON) format is a generic and widely-used non-biological file...
format and is popular for web-based applications or web-server asynchronous communication and data exchange. However, it is worth mentioning that Cytoscape.js (Franz et al., 2016) accepts JSON formats for network visualization. Finally, the Nexus and the Newick file formats are standard ways for representing trees. While NDEx (Pillich et al., 2017) is an open-source framework for the sharing of networks of many types and formats, file-format-specific parsers are available [e.g., Bioconductor (Gentleman et al., 2004) rBiopaxParser (Kramer et al., 2013), rBml, RPsiXML and others]. Examples of such file formats are shown in Figure 7.

**GRAPH LAYOUTS AND EDGE BUNDLING**

For graph analysis and interpretation, it is important to be able to depict a graph whose structure, symmetries, and other main features become clear in a visually and aesthetically appealing way. This is especially true for graphs of large size, where many nodes and edges can have multiple clusters and interconnected areas.

Graph drawing combines methods from mathematics and computer science to derive two- and three-dimensional representations of graphs, employing a number of strategies (Figure 8). Among the most successful layouts are the force-based layout approaches, where the nodes of the graphs are metaphorically modeled as point particles with attractive (spring) forces acting between nodes connected by an edge and repelling (electrical) forces acting between all pairs of nodes. The optimal layout is determined by the positions of nodes/particles that minimize the total energy of the system. Typically, such a state is found by simulating the forces of the many-particle physical system and arriving at a minimum energy state iteratively. In addition, in a spectral layout method, the coordinates are taken to be the eigenvectors of a matrix such as the Laplacian, derived from the adjacency matrix of the graph. Orthogonal layout
methods allow the edges of the graph to run horizontally or vertically, parallel to the coordinate axes of the layout, while tree layout algorithms use tree-like structures and are suitable for visualizing ontologies or hierarchies. Finally, circular layout methods place the vertices of the graph on a circle, choosing carefully the ordering of the vertices around the circle to reduce crossings and place adjacent vertices close to each other.

While graph drawing is a mature field with many proposed alternatives, the approaches that produce the most compelling visualizations (e.g., force directed based algorithms) can often become CPU and memory greedy and struggle with visualizing networks of more than a few thousands of nodes and edges. Empirical performance statistics can be found elsewhere (Pavlopoulos et al., 2017). Many layout algorithms are embedded in standard visualization tools: Gephi (Bastian et al., 2009) visualization tool comes with a great variety of algorithms such as OpenOrd (Martin et al., 2011) and Yifan-Hu (Yifan, 2005) force-directed algorithms. OpenOrd can layout networks consisting of over a million nodes in less than half an hour but aesthetics depends on the network's topology. The Yifan-Hu layout can give aesthetically comparable representations to the ones produced by the widely used but time-consuming Fruchterman-Reingold (Fruchterman and Reingold, 1991), with much faster performance. Other algorithms included in Gephi are the circular, contraction, dual circle, random, MDS, Geo, Isometric, GraphViz, and Force atlas layouts. Similarly, Cytoscape (Shannon et al., 2003) visualization tool comes with a rich variety of simple (grid, random, and circular) and more sophisticated (force-directed, hierarchical) layout algorithms. Finally, for more customized layouts, one can utilize the igraph library (Gabor and Nepusz, 2006). yWorks provides the professional software manufacturer with state-of-the-art diagramming components.

For even more aesthetic layouts, edge bundling methods can be utilized to provide significant clutter reduction and make visible high-level edge patterns clearer (Zhou et al., 2013). These methods are gaining ground over the years and are mostly divided in hierarchical or force directed. An overview of these methods is extensively described elsewhere (Zhou, 2016). Edge bundling methods are still computationally expensive and their main philosophy is to group edges together like a bundle of cables. Cytoscape and Tulip (Auber et al., 2017) are two of the most widely used visualization tools which have such methods incorporated. Basic node layout as well as edge-bundling examples are shown in Figure 8.

In general, force-directed layouts are very suitable for scale-free networks like PPIs or highly modular networks with distinct communities or high clustering coefficient. It would not make sense for example to apply a force-directed layout in a fully connected graph. Similarly, hierarchical layouts are more suitable for trees or tree-like graphs such as the Gene Ontology. Finally, it is worth mentioning that there is tradeoff against time, particularly because algorithms (e.g., layouts) grow time exponentially as the network increases.

**NETWORK VISUALIZATION**

Several techniques have been introduced for the visualization of networks varying from very simple (e.g., adjacency matrices) to more complex (e.g., force directed layouts in 2D or 3D). However, the selection of the appropriate visualization, highly depends on the type of network which needs to be visualized. For example, in a multi-Omics approach, one would like to see different types of information (e.g., proteomics, transcriptomics, metabolomics,
genomics) in a well-structured view. For this purpose, a multi-layered visualization would be much more preferable compared to a generic force directed layout. This way, nodes of different type are placed onto different layers, while connections are allowed both within a layer as well as across layers. In the case of multi-edge graphs, two bioentities can be connected in multiple ways. Two genes, for example, might be homologous, or neighbors in a genome or co-express in an experiment. STRING database is one of the most widely used databases which utilizes multi-edge graph visualization. In such networks, layouts can be applied taking into consideration only one connection type or any combination of them.

While force-directed or hierarchical visualizations are very common, they often fail in coping the so-called hairball effect (dense networks where all nodes are almost connected to any other node—no structure). To partially address this issue, circos and hive plots have been introduced. Hive plot views use “radially oriented” linear axes as a coordinate system. Nodes are placed on these axes and edges are drawn as curved links. While hive plots are general, they have been used in biology to successfully visualize cancer, gene-disease, and gene regulatory networks (Krzywinski et al., 2012). Similarly, Circos application (Krzywinski et al., 2009) enables a circular composition to show connections between nodes or positions, which are difficult to visually organize when the underlying layout is linear. Such plots are very widely used in biology to represent phenomena like genomic variations. Arc diagrams in which nodes are displayed along a single axis and links are represented with arcs, can be used for a similar purpose. Finally, bipartite graphs which are widely used in epidemiology and gene-disease networks need special visualization to show mutual relationships between the elements of their two collections. While several other visualization approaches can be applied on hierarchical graphs (e.g., Gene Ontology) and biochemical networks (e.g., pathways or petri nets), the most basic concepts are schematically shown in Figure 9.
GRAPH-BASED CLUSTERING

Clustering is the process of grouping a set of objects so that objects belonging in the same group (cluster) have similar properties. For this, many state-of-the-art algorithms take into account the network’s topology and try to cluster the network accordingly. For example, many approaches try to find densely connected areas in a network, others try to “break” the bridges (edges with high betweenness centrality) between distinct communities and others look for easiest flow paths or are based on node distances.

Despite the great variety of graph-based clustering algorithms available today (Xu and Wunsch, 2005; Brohée and van Helden, 2006; Moschopoulos et al., 2011), only few can cope with large-scale networks consisting of millions of nodes and edges. SPICI (Jiang and Singh, 2010) is one of the fastest algorithms and accepts as input a list of connections. It supports both dense and sparse matrices and tries to find local densely connected neighbors using heuristics. It has running time complexity $O(V \log V + E)$ time and needs $O(E)$ memory. It is not suitable for networks with many hubs and low clustering coefficient. Louvain (Blondel et al., 2008) on the other hand, is an old-fashioned but rather fast and greedy algorithm with $O(V \log V)$ time performance. Molecular Complex Detection (MCODE) (Bader and Hogue, 2003) is a widely-used algorithm in biology and very suitable for finding protein complexes in PPI networks. It has $O(Vd^3)$ time complexity where $d$ is the vertex size of the average vertex neighborhood in the input graph. Affinity-propagation (Frey and Dueck, 2007) detects ways that nodes in a network can exchange “messages” between each other very fast. It is a high-quality algorithm and comes with $O(V^2)$ time complexity. This might be a decent performance for medium-scale biological networks like gene co-expression or PPIs but not sufficient for larger networks like the literature-based or the knowledge-based ones. Markov Clustering (MCL) (Enright et al., 2002) is one of the mostly cited algorithms in the field and was initially introduced to detect protein families from sequence similarity networks. It uses random walks to detect highly-connected subgraphs using a mathematical bootstrapping procedure and is able to cluster a few million nodes in less than an hour. However, it is memory greedy, a bottleneck which has been solved with its parallel version HipMCL (Azad et al., 2018), a scalable distributed-memory implementation. HipMCL uses MPI (Forum, 1994), and OpenMP (Dagum and Menon, 1998) and can cluster a network consisting of 300 million nodes and $\sim$17 billion edges in only $\sim$6h using $\sim$136,000 cores.

While it is not in the scope of this review to go into each algorithm’s detail, we highly encourage readers to either try each

FIGURE 10 | Network clustering. (A) A Yeast PPI network. (B) The PPI network clustered with MCL. (C) The PPI network clustered with MCL with the initial connections restored. (D) The initial network structure with some MCL clusters highlighted. (E) A cluster in high resolution. (F) Gene Ontology enrichment related for the zoomed cluster. Visualization is offered through Cytoscape whereas clustering has been performed with the use of ClusterMaker2 plugin.
of them individually in their command line versions or through
the clusterMaker2 (Morris et al., 2011) Cytoscape plugin. In
Figure 10 for example, a Yeast PPI network (Gavin et al., 2006)
has been clustered with clusterMaker’s MCL algorithm, whereas
a randomly selected cluster has been annotated with Cytoscape’s
BiNGO plugin (Maere et al., 2005).

HIERARCHICAL CLUSTERING
Hierarchical clustering is a non-graph-based way of data
clustering which accepts a distance matrix containing all pairwise
distances between the nodes as input and outputs a dendrogram
showing the hierarchical relationship between the clusters. The
standard hierarchical algorithm has $O(n^3)$ time complexity of
and requires $O(n^2)$ memory, thus making this method inappropriate
for large data sets. Hierarchical clustering is divided in three
main categories. These are Single linkage which calculates the
smallest distance between objects in each iteration step, Complete
linkage which calculates the longest distance between objects in
each iteration step and Average linkage which uses the average
distance between all pairs of objects in every iteration step. For
more details, a survey explaining how hierarchical clustering
algorithms work and what are their variations can be found
elsewhere (Langfelder et al., 2008).

Notably, all calculations are based on a distance matrix (fully
connected graph) which can be generated by a correlation
matrix as $D_{ij} = 1 - PCC_{ij}$, $D$ is the distance matrix and
PCC a Pearson Correlation Matrix (e.g., gene co-expression
networks). Figure 11 shows an example of how five genes
can be hierarchically clustered according to their expression
values/patterns measured in three hypothetical conditions or
time points. The final output is a heatmap accompanied
by a dendrogram showing how genes are grouped together.
Notably, in cases where it is not straightforward which cutoff
to apply on the tree in order to define the number of
clusters, statistical methods to automate such task, are available
(Langfelder et al., 2008).

CLUSTERING COMPARISON
Different clustering algorithms or runs of the same algorithm
using different parameters can often lead to dissimilar results.
Therefore, it is essential to be able to compare different clustering
results between each other. This is especially useful when one
wants to compare the results of a clustering algorithm against
an “optimal” or desired clustering for example, to study an
algorithm’s accuracy.

For this purpose, several clustering comparison metrics have
been introduced. Generally speaking, these metrics can be
divided into three categories: (i) counting pairs, (ii) set overlaps,
or (iii) mutual information (Wagner and Wagner, 2007). Some
well-known clustering comparison metrics which are based on
counting pairs are the Chi Squared Coefficient (Mirkin, 2001), Rand Index (Rand, 1971), Fowlkes-Mallows Index (Fowlkes and Mallows, 1983), Mirkin Metric (or Equivalence Mismatch Distance), Jaccard Index and the Partition Difference (Li et al., 2004). Metrics based on set overlaps include the F-Measure (Fung et al., 2003), Meila-Heckerman & Maximum-Match-Measure (Marina Meil and David, 2001), and the Van Dongen-Measure (Dongen, 2000). Finally, clustering comparison metrics of the mutual information category include the Normalized Mutual Information by Strehl & Ghosh (Alexander and Joydeep, 2003), Normalized Mutual Information by Fred & Jain (Ana and Jain, 2003) and the Variation of Information methods (Meila, 2000).

The metrics of the first category count the number of object pairs that (a) were clustered together in both clusterings, (b) were clustered differently in both clusterings and (c) were clustered together in only one of the two clusterings. Rand Index is such a metric, ranging from 0 to 1 and is defined as: \[ \text{RandIndex}(C_1, C_2) = \frac{2(n_{ca}+n_{cb})}{n(n-1)} \] where \( C_1 \) and \( C_2 \) are two different clusterings of a data set with \( n \) objects, \( n_{ca} \) is the total number of object pairs that were clustered together in both clustering and \( n_{cb} \) the number of pairs that were clustered differently in both clusterings. Intuitively, the Rand Index calculates the fraction of same-clustered (together or separately) pairs against the number of all possible pairs and equals to 1 when all pairs are clustered in the same manner in both clusterings and to 0 when there is no pair clustered in the same manner in any of the two clusterings.

Metrics based on set overlaps try to map clusters between clusterings in accordance to their maximum overlap. The Meila-Heckerman measure compares the results of a clustering against the optimal clustering. This makes the method asymmetric, which means it cannot be used while comparing two clusterings without one being the optimal. The Maximum-Match-Measure is the symmetric version of this metric which iteratively looks for the largest element of the confusion matrix of the intersection values between all clusters of the two clusterings, meaning the cluster pair with the largest overlap. The column and row of the confusion matrix which contain the largest element are then crossed out and the sum of the results of all iterations are aggregated and divided by the total number of elements. The formula for the Maximum-Match-Measure is as follows:

\[ \text{MM}(C_1, C_2) = \frac{1}{n} \sum_{i=1}^{n} \max(k, l) \text{conf}^{i} \] where \( \text{conf}^{i} \) finishes in \( \min(k, l) \) steps, \( k \) and \( l \) are the respective numbers of clusters for clusterings \( C_1 \) and \( C_2 \) and \( \text{conf}^{i} \) is the confusion matrix described above with \( i - 1 \) columns and \( i - 1 \) rows being removed at each iteration. Maximum-Match-Measure ranges from 0 to 1.

An asymmetric and widely used clustering comparison metric of the set-overlap category is the F-Measure. The F-Measure indicates how close a clustering \( C_2 \) is to an optimal clustering \( C_1 \) by making use of the harmonic mean of precision and recall between each cluster, with precision \( p_{C_1|C_2} = \frac{\text{conf}_{i}}{n_{c_{1}^{*}}} \) and recall \( r_{C_1|C_2} = \frac{\text{conf}_{i}}{n_{c_{2}^{*}}} \), \( i \in [1, k] \) and \( j \in [1, l] \). The F-Measure between two clusters is calculated as \( F(C_1, C_2) = \frac{2 \cdot p_{C_1|C_2} \cdot r_{C_1|C_2}}{p_{C_1|C_2} + r_{C_1|C_2}} \) and the overall F-Measure between two clusterings is defined as the weighted sum of the maximum F-Measures for the clusters in \( C_2 \), \( F(C_1, C_2) = \sum_{i=1}^{k} \frac{n_{c_{1}^{*}}}{} \max_{l=1}^{l} F(C_{1i}, C_{2j}) \) and ranges in \( [0, 1] \).

Metrics of the mutual information clustering comparison category are based on the entropy of information and on the probability of finding an element in a specific cluster. The entropy of a clustering is defined as \( H(C) = -\sum_{i=1}^{k} P(i) \log_{2}p(i) \), where \( P(i) = \frac{n_{c_{i}^{*}}}{n_{c_{1}^{*}}} \) is the probability that a random...
element picked is a member of cluster $C_i$, and the mutual information between two clusterings $C_1$ and $C_2$ as $I(C_1, C_2) = \sum_{i=1}^{k} \sum_{j=1}^{l} P(i, j) \log \frac{P(i,j) P(C_1) P(C_2)}{P(C_1) P(C_2)}$, where $P(i,j) = \frac{\text{conf}_{ij}}{n}$ is the probability that an element belongs in cluster $C_i$ and also in $C_j$. The Variation of Information is a mutual information clustering comparison metric and is calculated as $VI(C_1, C_2) = H(C_1) + H(C_2) - 2I(C_1, C_2)$. Intuitively, the Variation of Information metric describes the amount of information we lose from the first clustering as well as the information we still have to gain from the second clustering. The Variation of Information metric is not bounded by a constant value but by a $\log(n)$ upper bound (in the case of two trivial clusterings).

Examples of Rand Index, Maximum-Match-Measure, F-Measure and Variation of Information are shown in Figure 12.

**NETWORK ALIGNMENT**

In today’s multi-Omics era, integration of heterogeneous information (e.g., transcriptomics, proteomics, metabolomics, etc.) in a multi-layered network structure is becoming a trend. Additionally, methods to directly compare networks and their topological features are gaining ground. To address these issues, network alignment, or alternatively graph isomorphism approaches can be used. Notably, graph alignment is not a trivial task as it is computationally expensive and has been characterized as NP-complete (Zampelli et al., 2010). The concept behind network alignment is to highlight conserved or missing nodes and edges across two (pairwise) or more (multiple) networks. In the biomedical field for example, an alignment could potentially be used for the discovery of conserved traits between different species (Sharan et al., 2005), the detection of common pathway interactions between two different disease states or the detection of deleted gene expression connections upon drug treatment. Like in a sequence alignment, a network alignment can also be either local or global.

Established implementations in the field include the NetworkBLAST aligner (Kalaev et al., 2008) for protein network alignment between two species or across multiple networks from different organisms, the MaWISh (Maximum Weight Induced Subgraph) (Koyutürk et al., 2006) for PPI alignments in order to underlie evolutionary relationships and the HGRAAL (Milenković et al., 2010) for metabolic networks of different species.

Until now, several graph alignment strategies have been introduced and various methods have been implemented. Some of the strategies are: modular graph kernels and divide and conquer strategies (Towfic et al., 2009), constraint programming (Zampelli et al., 2010), linear representation of networks (Kalaev et al., 2008), scoring functions (Flannick et al., 2008), connected-components (Tian and Samatova, 2008), heuristic searches (Kuchaiev et al., 2010), and graphlet degree vectors. Notably, recent developments allow the alignment of networks with multiple edge types (Gu et al., 2018).

While it is not in the scope of this article to cover all existing methods, for demonstration purposes, we present an example based on a simplified version of the GRAAL alignment method. The GRAAL network aligner takes into consideration the topology of a network and uses facets from both local and global alignment methods to produce a global alignment.

**FIGURE 13** | A topology-based network alignment example. (A) Possible graphlet compositions for 2 and 3 nodes. Orbits 0–3, which are annotated, represent the possible position for a node in the various graphlets. (B) G1 and G2 graph representations. (C) Graphlet degree signatures. The row names represent the nodes, while the column names the different orbits. (D) The final network alignment based on a simplified version of the GRAAL algorithm. Node $V_{1P}$ (in red color) remains unaligned.
According to the GRAAL algorithm, each node of the smaller network is aligned to exactly one node of the larger one. Let’s assume that there are two graphs $G_1 = (V_1, E_1)$ and $G_2 = (V_2, E_2)$. GRAAL introduces the concept of graphlets, which give a node a more detailed representation of its degree based on its local neighborhood of connections. All possible 2- and 3-node graphlet compositions are shown in Figure 13A. The aggregation of the number of 2-node graphlets attached to a node, represents the node’s degree. A node can only appear in one of the annotated orbits 0–3 in the respective graphlets in Figure 13A.

Figure 13B, demonstrates a network alignment example. During GRAAL’s first step, the lowest possible alignment cost for aligning each node from $G_1$ to each node of $G_2$ is calculated. Each row in Figure 13C represents the graphlet-signature for each node (for each graph, respectively), based on all possible orbits depicted in Figure 13A. The cost of aligning two nodes takes into consideration both their degrees and graphlet-signature similarity, whereas the lower cost is assigned to high degree nodes, as well as to graphlet similarities which do not replicate lower-degree graphlets. The starting seed node-pair with the lowest cost is chosen and the alignment is expanded outwards from these two nodes. In this example, nodes $V_2G$ and $V_1A$ have the highest degrees and the most similar graphlet-signatures, thus the pair $(V_2G, V_1A)$ is chosen as the first aligned seed. The next highest degree node connected to $V_2G$ is node $V_2F$. Node $V_2F$ is then randomly matched to one of the nodes $V_1B, V_1C, V_1D, V_1E$ (same degrees and similarity distances). For demonstration purposes let it be $V_1B$. Continuing on this graph’s path, node $V_2I$ is matched to $V_1C$. Moving on, node $V_2H$ is aligned to $V_1F$ and finally node $V_2K$ is randomly aligned to either $V_1D$ or $V_1E$. Let it be $V_1E$. $V_1D$ remains unaligned. The final GRAAL alignment is shown in Figure 13D.

**LINK PREDICTION**

Besides network alignment, predicting link changes in a single network has recently drawn attention in the biomedical field. Link prediction might concern the creation of future edges or the identification of missing links (e.g., incomplete data). While link prediction techniques are widely used by social media, in biological networks, they have also been used to identify potential drug side effects, protein-protein interactions, disease phenotypes based on molecular information and phylogenetic relations. For example, its application on bipartite graphs has unraveled new drug-target interactions (Kunegis et al., 2013). Its application on heterogeneous biological networks, has led to the identification of key pathway and protein interactions responsible for disease pathogenesis as well as candidate multiple sclerosis-associated genes (Himmelstein and Baranzini, 2015). Its combination with multi-way-based spectral has led to link prediction of protein–protein interaction networks.

The algorithm chosen for link prediction is often tied to the data type of the network. Due to the fact that each network type comes with its own growth pattern, relative assumptions must be made (Kunegis et al., 2013). Starting from an adjacency matrix $A$, through eigenvalue decomposition we can write $A$ as $U A U^T$. $U$ is an $n \times n$ orthogonal matrix and $A$ an $n \times n$ diagonal matrix. The values $A_{ii}$ are the eigenvalues of $A$, and the columns of $U$ are its eigenvectors. The spectral evolution model states that in dynamic networks, eigenvalues change over time while eigenvectors remain constant. Some of the most common link prediction algorithm categories are listed below. Triangle closing or triadic closure (Leskovec et al., 2008) is a method for predicting edges which will appear between nodes with common neighbors and widely used in social network analysis. Path counting (Lü et al., 2009) is the extension of triangle closing, giving two nodes with further level neighbors an additional, lower link prediction score. Graph kernels (Smola and Kondor, 2003; Ito et al., 2005) are functions which describe the similarity between two nodes and are often used for link prediction.

Here, we demonstrate a link prediction example based on the triangle closing model. The adjacency matrix $A$ of an unweighted and undirected network is shown in Figure 14A. The algebraic representation of the triangle closing model can be expressed

![Figure 14](image-url)
as a square $A^2$ of the adjacency matrix, where each $(i,j)$ pair contains the number of common neighbors between $i$ and $j$ at a given time $t$. Thus, $A^2_{ij} = \sum_k A_{ik}A_{jk}$. The $A^2$ matrix is shown in Figure 14B. Let the algorithm allow only one new edge creation at each iteration. At the time point $t+1$, the highest value of $A^2$ is 3 (between nodes V1 and V7). The new edge V1–V7 is created and the $A^2$ table is updated accordingly for the next iteration. If this edge already existed, the next highest value of $A^2$ would be checked. In case of duplicate highest values, the algorithm would have chosen one of the corresponding edges to create randomly. The network of this example and the new edge created at time point $t+1$ are depicted in Figure 14C.

Here, we present some tools which allow the computational link prediction on biological networks. HETIONET (Himmelstein and Baranzini, 2015) is an integrative biomedical knowledge network assembled from 29 different databases of genes, compounds, diseases, and more. Through HETIONET’s website (https://het.io/), researchers can browse an interactive biological network of 47,031 nodes (11 types) and 2,250,197 relationships (24 types) and formulate their own edge predictions. Linkpred (Guns, 2014) calculates the likelihood of potential edge-creation in a future snapshot of a network. There are 18 predictor functions (local and global) to choose from. LPmade is another link prediction software which specializes in link prediction via commonly used unsupervised link prediction methods such as Adamic/Adar, common neighbors, Jaccard’s coefficient, Katz, preferential attachment, PropFlow, rooted PageRank, SimRank, and weighted rooted PageRank.

**NETWORK PERTURBATION**

In biology, direct comparisons between a disease and a healthy state are very common, thus making the study of molecular changes essential. Therefore, at a network level, changes between such states are considered as biological network perturbations. In network medicine, a network’s topology can be used as the backbone to further predict side effects in a system even at a 65–80% success rate (Santolini and Barabási, 2018). In the same study for example, a topology-based methodology was applied on a chemotaxis network of bacteria in order to predict the dynamics of perturbations such as gene knockout and overexpression with 90% accuracy. Furthermore, gene editing techniques such as CRISPR/Cas9 also benefit from network perturbation studies as the combination of single-cell sequencing methods with CRISPR/Cas9 offers detailed information of gene-knockout effects at a cellular level (Holding et al., 2019). Similar to gene knockout, RNA interference (RNAi) is a protein silencing method, where RNA molecules inhibit gene expression by targeting their mRNA. Nested effect models (NEMs) constitute probabilistic graphical models that describe the directed hierarchical dependencies on a perturbation network. In a recent study (Siebourg-Polster et al., 2015), it has been shown that an extended version of NEMs (NEMix), was proposed and used on signaling pathways’ networks. In a use case scenario of a human rhinovirus (HRV) infection signaling network constructed from RNAi screening data, the proposed method inferred highly accurate signaling networks, fully aligned to the ones in KEGG database.

**TOOLS**

Nowadays, a great variety of tools for network storage, analysis, and interactive visualization is available. Recent review articles (Pavlopoulos et al., 2008a, 2013, 2015, 2017; Gehlenborg et al., 2010; O’Donoghue et al., 2010) discuss the main challenges in the field in terms of storage and scalability and highlight the advantages and shortcomings of the current state-of-the-art tools. Briefly, Cytoscape (Shannon et al., 2003), Cytoscape.js (Franz et al., 2016), Gephi (Bastian et al., 2009), Pajek (Mrvar and Batagelj, 2016), Ondex (Köhler et al., 2006), Proviz (Iragne et al., 2005), VisANT (Hu et al., 2009), Medusa (Pavlopoulos et al., 2011b), Osprey (Breitkreutz et al., 2002), Arena3D (Pavlopoulos et al., 2008b; Secrier et al., 2012), Graphia (Kajeka), and BioLayout Express (Theocharidis et al., 2009) are a state-of-the-art of the tools worth mentioning. While many of them are designed for general use, most of them can be used to tackle problems in the network biology field. Ondex for example can integrate heterogeneous data from various sources, Gephi, Pajek, and Graphia (https://kajeka.com/graphia/) are interactive visualizers suitable for large-scale networks, Cytoscape hosts a great variety (>200) of plugins (Saito et al., 2012) and focuses on network visualization and annotation, Arena3D visualizes stacks of 2D networks in 3D space using a multi-layer concept and BioLayout Express and Graphia (Kajeka) are designed for 3D visualizations. Most of these tools (e.g., Gephi, Cytoscape, and Graphia) are highly interactive and allow network editing (node/edge coloring, size changing, labeling, annotations, zooming/rotating, collapse/expand grouping, arrow types, node/edge filtering, etc). In addition, tools such as the Network Analyzer (Doncheva et al., 2012), ZoomOut (Athanasiadis et al., 2015), Network Analysis Toolkit (NEAT) (Brohée et al., 2008), and NAP (Theocharidis et al., 2017) focus on the topological analysis, whereas non-interactive libraries such as the Stanford Network Analysis Project (SNAP) (Leskovec and Sosić, 2016), the outdated Large Graph Layout (LGL) (Adai et al., 2004), NetworkX (Hagberg et al., 2008), and GraphViz are command line applications able to offer back-end calculations as well as static visualizations. Specialized tools such as the Pathview (Luo and Brouwer, 2013), BioTapestry (Longabaugh, 2012), PathVisio (Kutmon et al., 2015), Interactive Pathways Explorer (iPath) (Darzi et al., 2018), MapMan (Thimm et al., 2004), and WikiPathways (Slenter et al., 2018), Pathway Commons (Rodchenkov et al., 2019) are designed for pathway analysis and visualization and finally, tools such as the Dendroscope (Huson et al., 2007) and iTOL (Letunic and Bork, 2007) are widely-used tree visualizers (Pavlopoulos et al., 2010). GeneMANIA (Franz et al., 2018) is offered for the detection of genes that are related to a set of input genes, using a very large set of functional association data (protein and genetic interactions, pathways, co-expression, co-localization, and protein domain similarity).
Concerning libraries, igraph (Gabor and Nepusz, 2006) is an open source library for network analysis and can be used by both Python and R languages. It offers a very rich plethora of functions dedicated to network analysis while it emphasizes on efficiency, portability, and ease of use. VisNetwork is a JavaScript-based R package for network visualization and ggplot2 an R data visualization package suitable for interactive charts and plots. Graphviz is an open source graph visualization software for representing structural information such as diagrams of abstract graphs and networks. NetworkX is a D3 JavaScript library and plotly a library suitable for data analytics. In addition, Three.js is a cross-browser JavaScript library for animated 3D computer graphics in a web browser and ndtv-d3 a library suitable for timelines and animated movies of objects. Plotly is a library suitable for timelines and animated movies of objects. NetworkX is a Python package for the creation, manipulation, and study of abstract graphs and networks. Graph-tool (Peixoto, 2017) a Python module for manipulation and statistical analysis of graphs. Finally, as biological network analyses become more and more popular, data exchange is crucial. For this purpose, the NDEx Project (Pillich et al., 2017) provides an open-source framework where scientists and organizations can share, store, manipulate, and publish biological network knowledge.

DISCUSSION

The adoption of mature high-throughput -Omics approaches to analyze biological samples (e.g., genomics, transcriptomics, proteomics, metabolomics etc.) has led to the production of data at the scale of tera- to peta-byte in size. Subsequently, due to this trend, biological networks follow an exponential growth, thus making their exploration, visualization, analysis, and storage a very difficult task. Therefore, traditional algorithms and data structures often fail to address scalability issues, thus making the adoption of modern technologies a necessity. As current tools are often limited in coping with large-scale datasets, Big Data approaches as well as parallel processing could be used for storing, querying, and processing large data volumes. Ideally, network analysis and visualization software could support algorithms which can run on distributed memory or multiple CPU and GPU systems for increased performance. In addition, global state-of-the-art data structures adjusted to such systems would be of great benefit.

Another bottleneck in systems biology is the visualization and representation of large-scale networks. As networks increase in size and complexity, more efficient algorithms for visualization are necessary. Notably, an alternative way to overcome 2D/3D space limitations is the adoption of virtual reality (VR) technologies. This way, biological networks could be for example explored or browsed using virtual universes. Typical examples for visualizing living systems such as a whole cell using such technology are the Visible Cell (Gagescu, 2001) or CELLmicrocosmos (Sommer, 2019). However, even after a decade of its existence, graphical limits, and cost of VR devices are still restrictive factors to be considered.

AUTHOR CONTRIBUTIONS

MK and EK wrote most of the manuscript. DP-E helped with the structure and with the biological content. GP conceived the concept and supervised the whole process.

FUNDING

Supported by the Operational Program Competitiveness, Entrepreneurship and Innovation, NSRF 2014-2020, Action code: MIS 5002562, co-financed by Greece and the European Union (European Regional Development Fund). EK has been partially supported by the Action Strengthening Human Resources, Education and Lifelong Learning, 2014–2020, co-funded by the European Social Fund (ESF) and the Greek State.

ACKNOWLEDGMENTS

We thank Dr. Yorgos Sofianatos for his valuable input and feedback.

REFERENCES

Adai, A. T., Date, S. V., Wieland, S., and Marcotte, E. M. (2004). LGL: creating a map of protein function with an algorithm for visualizing very large biological networks. J. Mol. Biol. 340, 179–190. doi: 10.1016/j.jmb.2004.04.047

Al-Anzi, B., Arpp, P., Gerges, S., Ormerod, C., Olsman, N., and Zinn, K. (2015). Experimental and computational analysis of a large protein network that controls fat storage reveals the design principles of a signaling network. PLoS Comput. Biol. 11:e1004264. doi: 10.1371/journal.pcbi.1004264

Alexander, S., and Joydeep, G. (2003). Cluster ensembles—a knowledge reuse framework for combining multiple partitions. J. Mach. Learn. Res. 3, 583–617. doi: 10.1162/153244303321897735

Altschul, S. F., Gish, W., Miller, W., Myers, E. W., and Lipman, D. J. (1990). Basic local alignment search tool. J. Mol. Biol. 215, 403–410. doi: 10.1006/jmbi.1990.1204

Ana, L. N. F., and Jain, A. K. (2003). “Robust data clustering,” in: IEEE Computer Society Conference on Computer Vision and Pattern Recognition, 2003 Proceedings (Madison, WI: IEEE Comput. Soc.), II–128 II–133. Available online at: http://ieeexplore.ieee.org/abstract/document/1211462/ (accessed August 21, 2019).

Anathasios, E. I., Bourdakou, M. M., and Spyrou, G. M. (2015). Zoomout: analyzing multiple networks as single nodes. IEEE/ACM Trans. Comput. Biol. Bioinform. 12, 1213–1216. doi: 10.1109/TCBB.2015.2424411

Auber, D., Archambault, D., Bourqui, R., Delest, M., Dubois, J., Lambert, A., et al. (2017). “Tulip 5,” in Encyclopedia of Social Network Analysis and Mining, eds R. Alhajj and J. Rokne (New York, NY: Springer New York), 1–28.

Azad, A., Pavlopoulos, G. A., Ouzounis, C. A., Kyprides, N. C., and Buluç, A. (2018). HipMCL: a high-performance parallel implementation of the Markov clustering algorithm for large-scale networks. Nucleic Acids Res. 46:e33. doi: 10.1093/nar/gkx1313

Bader, G. D., Betel, D., and Hogue, C. W. (2003). BIND: the biomolecular interaction network database. Nucleic Acids Res. 31, 248–250. doi: 10.1093/nar/gkg056

Bader, G. D., and Hogue, C. W. (2003). An automated method for finding molecular complexes in large protein interaction networks. BMC Bioinform. 4:2. doi: 10.1186/1471-2105-4-2

Barabasi, A.-L., and Albert, R. (1999). Emergence of scaling in random networks. Science 286, 509–512. doi: 10.1126/science.286.5439.509
Gene Ontology Consortium (2004). The gene ontology (GO) database and informatics resource. *Nucleic Acids Res.* 32, 258D–261D. doi: 10.1093/nar/gkh036

Gentleman, R. C., Carey, V. J., Bates, D. M., Bolstad, B., Dettling, M., Dudoit, S., et al. (2004). Bioconductor: open software development for computational biology and bioinformatics. *Genome Biol.* 5:S80. doi: 10.1186/gb-2004-5-10-s80

Gioulakis, A., Klapa, M. L. and Moschonas, N. K. (2017). PICKLE 2.0: a human protein-protein interaction meta-database employing data integration via genetic information ontology. *PLoS ONE* 12:e0186039.

Goh, K.-L., Cusick, M. E., Valle, D., Childs, B., Vidal, M., and Barabasi, A.-L. (2007). *Nature* 441, 767–776. doi: 10.1038/nature04795

Hajduckova et al. Biological Network Analysis

Hagberg, A., Schult, D., and Swart, P. (2008). "Exploring network structure, dynamics, and function using network," in *Proceedings of the 7th Python in Science Conference* (Pasadena, CA: SciPy), 11–15.

Hage, P., and Harary, F. (1995). Eccentricity and centrality in networks. *So. Netw.* 17, 57–63. doi: 10.1016/0387-8733(94)00248-9

Han, H., Cho, J.-W., Lee, S., Yun, A., Kim, H., Bae, D., et al. (2018). TRUST v2: an expanded reference database of human and mouse transcriptional regulatory interactions. *Nucleic Acids Res.* 46, D380–D386. doi: 10.1093/nar/gkx1013

Han, H., Lee, S., and Lee, I. (2019). NGSEA: network-based gene set enrichment analysis for interpreting gene expression phenotypes with functional gene sets. *Mol. Cells* 42, 579–588. doi: 10.1007/s10059-019-00349-1

Hermjakob, H., Montecchi-Palazzi, L., Bader, G., Alfarano, C., Helmus, M., Zvelebil, M., and Cesareni, G. (2004). BioPST: protein similarity through the alignment of databases. *Nucleic Acids Res.* 32, 27–30. doi: 10.1093/nar/gkh086

Hermjakob, H., Montecchi-Palazzi, L., Zucchelli, G., Cusin, D., Cesareni, G., and Bader, G. (2004). *Nature* 432, 772–776. doi: 10.1038/nature03093

Huson, D. H., Richter, D. C., Rausch, C., Dettli, T., Franz, M., and Rupp, R. (2007). Dendroscope: an interactive viewer for large phylogenetic trees. *BMC Bioinform.* 8:460. doi: 10.1186/1471-2105-8-460

Huson, D. H., Rupp, R., and Scornavacca, C. (2010). *Phylogenetic Networks: Concepts, Algorithms and Applications.* Cambridge: Cambridge University Press. Available online at: http://ebooks.cambridge.org/ref/id/CBO9780511970476

Hung, J.-H., Yang, T.-H., Hu, Z., Weng, Z., and DeLisi, C. (2012). Gene set enrichment analysis: performance evaluation and usage guidelines. *Brief Bioinform.* 13, 281–291. doi: 10.1093/bib/bbr049

Huson, D. H., Richter, D. C., Rausch, C., Dettli, T., Franz, M., and Rupp, R. (2007). Dendroscope: an interactive viewer for large phylogenetic trees. *BMC Bioinform.* 8:460. doi: 10.1186/1471-2105-8-460

Ings, T. C., Montoya, J. M., Bascompte, J., Blüthgen, N., Brown, L., Dormann, C. F., et al. (2009). *Ecological networks—beyond food webs.* *J. Anim. Ecol.* 78, 253–269. doi: 10.1111/j.1365-2668.2008.01460.x

Iragne, F., Nikolosi, M., Mathieu, A., Audier, D., and Sherman, D. (2005). ProViz: protein interaction visualization and exploration. *Bioinform. Oxf. Engl.* 21, 272–274. doi: 10.1093/bioinformatics/bth455

Ito, T., Shimbo, M., Kudo, T., and Matsumoto, Y. (2005). "Application of kernels to link analysis," in *Proceeding of the Eleventh ACM SIGKDD International Conference on Knowledge Discovery in Data Mining* - KDD’05 (Chicago, IL: ACM Press). 586. Available online at: http://portal.acm.org/citation.cfm?doid=1081780.1081941 (accessed December 17, 2019).

Jalili, M., Salehzadeh-Yazdi, A., Gupta, S., Wolkenhauer, O., Yaghmaie, M., Resendis-Antonio, O., et al. (2016). Evolution of centrality measurements for detecting the presence of essential proteins in biological networks. *Front. Physiol.* 7:375. doi: 10.3389/fphys.2016.00375

Jaeger, H., Tombor, B., Albert, R., Oltvai, Z. N., and Barabasi, A. L. (2000). The large-scale organization of metabolic networks. *Nature* 407, 651–654. doi: 10.1038/350395

Jiang, P., and Singh, M. (2010). SPICi: a fast clustering algorithm for large biological networks. *Bioinform. Oxf. Engl.* 26, 1105–1111. doi: 10.1093/bioinformatics/btq078

Junker, B. H., and Schreiber, F. (eds) (2008). *Analysis of Biological Networks.* *Wiley Series on Bioinformatics* (Hoboken, NJ: Wiley-Interscience). 346. doi: 10.1002/0470072534

Kalaev, M., Smoot, M., Ideker, T., and Sharan, R. (2008). NetworkBLAST: comparative analysis of protein networks. *Bioinformatics* 24, 594–596. doi: 10.1093/bioinformatics/btm630

Kandasamy, K., Moman, S., Raju, R., Keerthikumar, S., Kumar, G. S. S., Venugopal, A. K., et al. (2010). NetPath: a public resource of curated signal transduction pathways. *Genome Biol.* 11:R3. doi: 10.1186/gb-2010-11-1-r3

Kanehisa, M., and Goto, S. (2000). KEGG: kyoto encyclopedia of genes and genomes. *Nucleic Acids Res.* 28, 27–30. doi: 10.1093/nar/28.1.27

Kashtan, N., Itzkovitz, S., Milo, R., and Alon, U. (2004). Efficient sampling algorithm for estimating subgraph concentrations and detecting network motifs. *Bioinform. Oxf. Engl.* 20, 1746–1758. doi: 10.1093/bioinformatics/bth163

Kavarucu, Y. (2015). A comparative study on network motif discovery algorithms. *Int. J. Data Min. Bioinform.* 11, 180–204. doi: 10.1504/IJDM.2015.066777

Klibas, S. M., Wan, R., Sato, K., Horton, P., and Frith, M. C. (2011). Adaptive seeds tame genomic sequence comparison. *Genome Res.* 21, 487–493. doi: 10.1101/gr.119853.110

Kim, W., Li, M., Wang, J., and Pan, Y. (2011). Biological network motif detection and evaluation. *BMC Syst. Biol.* 5 (Suppl. 3):S5. doi: 10.1186/1752-0509-5-S3-S5

Kirch, W. (ed) (2008). *Pearson’s Correlation Coefficient.* In: *Encyclopedia of Public Health* (Dordrecht: Springer Netherlands), 1090–1. Available online at: http://link.springer.com/10.1007/978-1-4020-5614-7_2569 (accessed November 12, 2019).

Knuth, D. E. (1997). *The Art of Computer Programming.* 3rd Edn. Reading: Addison-Wesley.

Köhler, I., Raabmacht, J., Taubert, J., Specht, M., Skusa, A., Rüegg, A., et al. (2006). Graph-based analysis and visualization of experimental results with ONDEX. *Bioinform. Oxf. Engl.* 22, 1383–1390. doi: 10.1093/bioinformatics/btl081

Koschützki, D., and Schreiber, F. (2008). Centrality analysis methods for biological networks and their application to gene regulatory networks. *Gene Regul. Syst. Biol.* 2, 193–201. doi: 10.4137/GRSB.5702

Koytik, M., Kim, Y., Topkara, V., Subramanian, S., Szpankowski, W., and Grama, A. (2006). Pairwise alignment of protein interaction networks. *J. Comput. Biol.* 13, 182–199. doi: 10.1089/cmb.2006.13.182

Kramer, F., Bayerlová, M., Klemm, F., Bleekmann, A., and Beissbarth, T. (2013). rBiopaxParser—an R package to parse, modify and visualize BioPAX data. *Bioinform. Oxf. Engl.* 29, 520–522. doi: 10.1093/bioinformatics/bts710

Koutroul et al. Biological Network Analysis
metabolomics to other omics research. *Nucleic Acids Res.* 46, D661–D667. doi: 10.1093/nar/gkx1064

Smith, T. F., and Waterman, M. S. (1981). Identification of common molecular subsequences. *J. Mol. Biol.* 147, 195–197. doi: 10.1016/0022-2836(81)90087-5

Snola, A. J., and Kondor, R. (2003). "Kernels and regularization on graphs," in *Learning Theory and Kernel Machines*, eds B. Schölkopf and M. K. Warmuth (Berlin: Springer Berlin Heidelberg), 144–158. Available online at: http://link.springer.com/10.1007/978-3-540-45167-9_12 (accessed December 17, 2019).

Sommer, B. (2019). The CELLC microcosmos tools: a small history of java-based cell and membrane modelling open source software development. *J. Integr. Bioinform.* 16:20190057. doi: 10.1515/jib-2019-0057

Sonawane, A. R., Weiss, S. T., Glass, K., and Sharma, A. (2019). Network medicine in the age of biomedical big data. *Front. Genet.* 10. doi: 10.3389/fgene.2019.00294

Spriń, V., and Mirny, L. A. (2003). Protein complexes and functional modules in molecular networks. *Proc. Natl. Acad. Sci. U.S.A.* 100, 12123–12128. doi: 10.1073/pnas.2033241400

Stark, C., Breitkreutz, B.-J., Reguly, T., Boucher, L., Breitkreutz, A., and Tyers, M. (2006). BioGRID: a general repository for interaction datasets. *Nucleic Acids Res.* 34, D535–D539. doi: 10.1093/nar/gkj109

Stone, L., Samberloff, D., and Artzy-Randrup, Y. (2019). Network motifs and their origins. *PLoS Comput. Biol.* 15:e1006749. doi: 10.1371/journal.pcbi.1006749

Subramanian, A., Tamayo, P., Mootha, V. K., Mukherjee, S., Ebert, B. L., Gillette, M. A., et al. (2005). Gene set enrichment analysis: a knowledge-based approach for interpreting genome-wide expression profiles. *Proc. Natl. Acad. Sci. U.S.A.* 102, 15545–15550. doi: 10.1073/pnas.0506581012

Szklarczyk, D., and Jensen, L. I. (2015). "Protein-protein interaction databases," in *Protein-Protein Interactions*, eds C. L. Meyerkord and H. Fu (New York, NY: Springer New York), 39–56.

Szklarczyk, D., Santos, A. von Mering, C., Jensen, L. J., Bork, P., and Kuhn, M. (2016). STITCH 5: augmenting protein-chemical interaction networks with tissue and affinity data. *Nucleic Acids Res.* 44, D380–D384. doi: 10.1093/nar/gkv1277

Theodoridis, A., van Dongen, S., Enright, A. J., and Freeman, T. C. (2009). Network visualization and analysis of gene expression data using biolayout express(3D). *Proc. Natl. Acad. Sci. U.S.A.* 106, 294–307. doi: 10.1073/pnas.0901885106

Yang, Z. (1996). Phylogenetic analysis using parsimony and likelihood methods. *J. Mol. Evol.* 42, 294–307. doi: 10.1007/BF02198856

Yeşilbaş, I., Sharipov, R., Kolmykov, S., Kondrakhin, Y., and Kolpakov, F. (2019). GTRGD: a database on gene transcription regulation-2019 update. *Nucleic Acids Res.* 47, D100–D105. doi: 10.1093/nar/gky1128

Yifan, H. (2005). Efficient, high-quality force-directed graph drawing. *Math. Prog.* 10, 37–77. Available online at: http://ausm.lyds.mfe5343/paper/27%20High-Quality%20Force-Directed%20Graph%20Drawing.pdf

Yu, X., Wang, Z., Huang, J., Putharasathy, S., Mosavinasab, S., Huang, Y., et al. (2019). Graph embedding on biomedical networks: methods, applications and evaluations. *C. Wright. Sci. Technol.* 18, 145–156. doi: 10.1109/TSVT.2013.6590998

Conflict of Interest: The authors declare that the research was conducted in the absence of any commercial or financial relationships that could be construed as a potential conflict of interest.

Copyright © 2020 Koutrouli, Karatzas, Paez-Espin and Pavlopoulos. This is an open-access article distributed under the terms of the Creative Commons Attribution License (CC BY). The use, distribution or reproduction in other forums is permitted, provided the original author(s) and the copyright owner(s) are credited and that the original publication in this journal is cited, in accordance with accepted academic practice. No use, distribution or reproduction is permitted which does not comply with these terms.