Scalable mining, analysis and visualisation of protein-protein interaction networks

Shaikh Arifuzzaman* and Bikesh Pandey

Department of Computer Science,
University of New Orleans,
New Orleans, LA 70148, USA
Email: smarifuz@uno.edu
Email: bpandey@uno.edu
*Corresponding author

Abstract: Proteins are linear chain biomolecules that are the basis of functional networks in all organisms. Protein-protein interaction (PPI) networks are networks of protein complexes formed by biochemical events and electrostatic forces. PPI networks can be used to study diseases and discover drugs. The causes of diseases are evident on a protein interaction level. For instance, elevation of interaction edge weights of oncogenes is manifested in cancers. The availability of large datasets and need for efficient analysis necessitate the design of scalable methods leveraging modern high-performance computing (HPC) platforms. In this paper, we design a lightweight framework on a distributed-memory parallel system to study PPI networks. Our framework supports automated analytics based on methods for extracting signed motifs, computing centrality, and finding functional units. We design message passing interface (MPI)-based parallel methods and workflow, scalable to large networks. To the best of our knowledge, these capabilities collectively make our tool novel.

Keywords: protein interaction; biological networks; network visualisation; massive networks; HPC systems; network mining.

Reference to this paper should be made as follows: Arifuzzaman, S. and Pandey, B. (2019) 'Scalable mining, analysis and visualisation of protein-protein interaction networks', *Int. J. Big Data Intelligence*, Vol. 6, Nos. 3/4, pp.176–187.

Biographical notes: Shaikh Arifuzzaman is an Assistant Professor of Computer Science at the University of New Orleans (UNO), USA. His research interests are in big data analytics, parallel algorithms, network science, and high-performance computing. Earlier, he obtained his PhD in Computer Science from Virginia Tech where he worked in Network Dynamics and Simulation Science Lab (NDSSL). He also worked in Data Sciences and Cyber Analytics Department at Sandia National Laboratories, CA, USA. At UNO, he leads the big data and scalable computing research group and currently works on parallel graph algorithms, big data security, and several data-intensive interdisciplinary problems.

Bikesh Pandey graduated from the University of New Orleans with a major in Computer Science with a concentration in Game Development and a minor in Mathematics. His research included research on large protein networks, image processing and designing algorithms. Prior to this paper, he has worked on designing motion tracking system that can track movements across lenses.

This paper is a revised and expanded version of a paper entitled 'Scalable mining and analysis of protein-protein interaction networks' presented at 3rd IEEE International Conference on Big Data Intelligence and Computing (DataCom), Orlando, FL, 6–10 November 2017.

1 Introduction

Studying network (graph) data is fundamental in diverse scientific disciplines since network is a powerful abstraction for representing interactions among entities in a system (Newman, 2003; Girvan and Newman, 2002). The entities and their interactions are represented as nodes (vertices) and links (edges) of a network, respectively. Examples include biological networks (Girvan and Newman, 2002; Chen and Lonardi, 2009), the web graph (Broder et al., 2000), various social networks (Kwak et al., 2010), and many other information networks. Mining biological data is of growing interest since they represent fundamental bio-chemical mechanisms in a cell or in a living organism (Chen and Lonardi, 2009). Due to the advancement of data and computing technology, biology and related disciplines generate a large volume and variety of data (Girvan and Newman, 2002), many of them are about proteins and protein-protein interactions (PPI) (Ewing et al., 2007). PPI networks offer an excellent chance to study disease dynamics in molecular level and shed light on drug

discovery (Bader et al., 2004; Han et al., 2004; Schwikowski et al., 2000). However, large volume and variety of PPI datasets pose computational challenges, which motivates for scalability, both in algorithmic methods and analysis workflow. In this paper, we develop an HPC-based framework to apply network-centric approaches to study PPI networks.

1.1 Studying PPI networks: significance and relevance

Proteins are linear chain biomolecules that are the basis of functional networks in all organisms. Aspects of their interactions are of growing interest (Rual et al., 2005; Stelzl et al., 2005). PPI networks can be used to study disease and for drug discovery (Altieri, 2008; Brastianos et al., 2015). They also reveals the causes of diseases – for instance, most cancers are caused by increasing interaction edge weights of oncogenes and decreasing interaction edge weights of tumour suppressor genes (Altieri, 2008; Chin et al., 2004). Most human diseases are thought to have fewer than five causal PPIs; many have two or fewer causal interactions (Hopkins, 2008). Further, PPI networks help in drug discovery. Many approved drugs target a particular PPI (Altieri, 2008; Hopkins, 2008).

PPI networks have been shown to be relevant to treatment of diseases and drug discovery (Altieri, 2008; Hopkins, 2008). Further, there has been a line of work focusing on purely the bio-chemical aspect of PPIs. Unlike those works, this paper stresses on computing (mining and analysis) aspect of knowledge discovery and demonstrate how we can relate our results to biochemical contexts. There have been earlier works suggestive of the effectiveness of network-based approaches for analysing PPIs (Altieri, 2008; Hopkins, 2008). Local and global PPI network structural motifs suggest therapeutic strategies. The centrality hub nodes of PPI networks can be good candidates for drug target. Works such as (Altieri, 2008) use both global PPI information and pathway knowledge to reveal more bio-chemical insights. Most work related to network analysis do not consider signed and weighted networks (Suri and Vassilvitskii, 2011; Chiba and Nishizeki, 1985). However, PPI networks are both signed and weighted. Moreover, many existing methods are not scalable to large networks (Fortunato and Lancichinetti, 2009; http://vlado.fmf.uni-lj.si/pub/networks/pajek/; https://networkx.github.io/). Scalable parallel and sampling-based algorithms (Arifuzzaman and Khan, 2015; Arifuzzaman et al., 2013, 2015a, 2015b) are required to deal with large network data.

1.2 Unique challenges for scalable analysis

In the era of big data, we are deluged with network data from a wide range of areas. The volume of biological and bio-medical data is also growing rapidly. The string repository (https://string-db.org/) has PPI networks with 9.6M proteins and 1,380M interactions. There are many other public repositories (https://thebiogrid.org/) that share large biological datasets. This emergence of large-scale network data motivates us to find scalable algorithms and tools for extracting useful intelligence. In some cases, these networks do not fit into the main memory of a single computing node. Further, an algorithm having a high computational complexity might fail to work on networks with a few millions edges.

1.3 Contributions of this work

In this paper, we describe our framework for highly scalable and rigorous methods for mining and analysing PPI networks. To address the issues emerged from large-scale datasets, we develop a workflow consisting of scalable labelled graph analysis algorithms leveraging large distributed multi-core clusters. The key contributions are as follows.

1 *A high performance computing-based tool that scales:* the tool includes scalable parallel methods (algorithms) for discovering functional units and extracting motifs in PPI networks. Our methods and workflow scale to large networks for a wide variety of network metrics.

2 *An extensible framework that can integrate new methods:* the tool currently includes many mining and analysis methods including counting triangular motifs, community detection, computing diameter, and several centrality and path-based metrics. Any new methods can easily be integrated with the tool.

3 *Identification of relevance to biological or bio-medical contexts of PPI:* our methods for signed motif extraction, centrality computation, and discovery of functional units can be used to identify target proteins and important hubs. Such network motifs and properties of a PPI network have useful implications for drug target discovery.

4 *Promotion of interdisciplinary collaboration:* we anticipate this tool can facilitate multidisciplinary investigations consisting of experts from both computational and biological domains. Further, the tool can essentially be generalised to other related applications in neuroscience, medical informatics, and likes.

The rest of the paper is organised as follows. The datasets and computing resources are briefly described in Section 2. We present the overview and architecture, capabilities, and evaluation of our framework in Sections 3, 4 and 5, respectively. We compare our tool with existing network analysis tools in Section 6. We conclude in Section 7.

2 Preliminaries

We present our datasets, computational model, and resources below.

2.1 Notation and definitions

The given network is denoted by $G(V, E)$, where V and E are the sets of vertices and edges, respectively, with $m = |E|$ edges and $n = |V|$ vertices labelled as 0, 1, 2, …, $n - 1$. We use the words *node* and *vertex* interchangeably. We assume that the input network is undirected. If $(u, v) \in E$, we say u and v are neighbours to each other. The set of all neighbours of $v \in V$ is denoted by $\mathcal{N}_v$, i.e., $\mathcal{N}_v = \{u \in V \mid (u, v) \in E\}$. The degree of v is $d_v = |\mathcal{N}_v|$.

We will also introduce notations in later sections when required. We use several network analysis algorithms including counting triangles. A triangle is a set of three nodes $u, v, w \in V$ such that there is an edge between each pair of these three nodes, i.e., (u, v); (v, w), $(w, u) \in E$. The number of triangles incident on v, denoted by T_v, is same as the number of edges among the neighbours of v, i.e.,

$$T_v = \left|\{(u, w) \in E : u, w \in \mathcal{N}_v\}\right|.$$

To discuss the distributed-memory system we use, let P be the number of processors used in the computation, which we denote by $p_0, p_1, \ldots, p_{P-1}$ where each subscript refers to the rank of a processor.

We use K, M and B to denote thousands, millions and billions, respectively; e.g., 1B stands for one billion.

2.2 Datasets

We study PPI networks from StringDB database (https://string-db.org/) for several organisms. The networks are represented as edgelists with several interaction values based on various evidences such as interaction and coexpression scores. The datasets we use are summarised in Table 1. These datasets contain an edge weight valued on a scale of 0–1,000 between two proteins. This weight is the overall interaction score – sum of all the categorical scores such as coexpression score, neighbourhood score, experimental score, and several other values given by the database. The datasets identify proteins using unique protein identifiers called Ensembl Protein IDs determined by Ensembl.org. Further details on these proteins and also other genes can also be found at Ensembl Genome Browser (http://www.ensembl.org).

We also experimented on other datasets found from National Center for Biotechnology Information (https://www.ncbi.nlm.nih.gov/genome/viruses/retroviruses/hiv-1/interactions/browse/) and BioGrid (https://thebiogrid.org/). Many of these datasets have no quantifiable interaction scores that can be further analysed. Even though we experimented on datasets from several sources, many of them are not presented in this paper for brevity.

2.3 Computation model and resources

The parallel algorithms our tool uses were developed for message passing interface (MPI)-based distributed-memory parallel systems. Each processor has its own local memory. The processors do not have any shared memory, and they communicate via exchanging messages. Compute resources are the physical resources on which individual jobs are executed. Our current resources include two HPC Linux clusters at Louisiana Optical Network Infrastructure (LONI) (https://loni.org/) and our host institution. Loni QueenBee system is a 50.7 TFlops peak performance 680 compute node cluster running the Red Hat Enterprise Linux 4 operating system. Each node contains two Quad Core Xeon 64-bit processors operating at a core frequency of 2.33 GHz. The compute cluster at our host institution is a small cluster with two large-memory computing nodes, each with 16 cores and 512 GB of RAM, connected by QDR infiniband interconnect and running Linux operating system.

Potential compute resources include any traditional HPC clusters, compute grids, clouds (e.g., Amazon Web Services), or dedicated servers with MPI libraries. A typical compute resource runs our network analysis kernels. Our middleware (control unit) and data and visualisation units also run on compute resources.

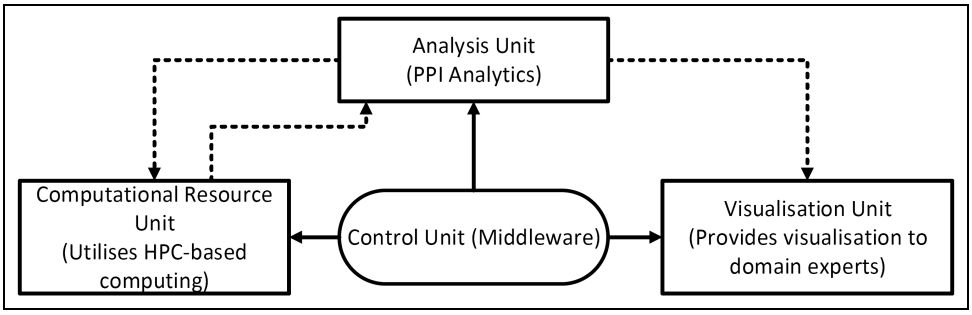
3 New generation graph analytical tool for PPI Networks

The use of network (graph) analysis for understanding protein interactions and their implication on broader aspects of biological process in organisms is still nascent (Altieri, 2008; Hopkins, 2008), and more studies are needed to demonstrate a clearer picture of results. In this paper, we hope to contribute to this literature by developing an HPC-based tool that helps assess both node and clustering-based characterisation of PPI networks.

Table 1 A subset of datasets used in our experiments

Network	*Nodes*	*Edges*	*Source*
Homo Sapiens	19,247	4,274,001	StringDB (https://string-db.org/)
Acetobacterium Woodii	3,439	369,956	StringDB (https://string-db.org/)
Albugo Laibachii	5,849	1,443,060	StringDB (https://string-db.org/)
Dinoroseobacter Shibae	3,567	412,618	StringDB (https://string-db.org/)
Bacillus Cytotoxicus	3,765	298,873	StringDB (https://string-db.org/)

Figure 1 Architectural overview of our framework for scalable mining, analysis, and visualisation of PPI networks



The presented tool is a first on several levels. The framework builds upon and extends significantly the existing work on scalable algorithms for graph data preprocessing (Arifuzzaman and Khan, 2015), counting triangular motifs (Arifuzzaman et al., 2013), and efficient parallel load balancing schemes (Arifuzzaman et al., 2015a). It complements the protein interaction literature with scalable algorithmic methods for efficient analysis. It is well established that causation of disease and drug discovery have significant correlation with network properties of nodes in PPI networks (Stelzl et al., 2005; Altieri, 2008; Hopkins, 2008).

Based on the prior work of the authors on network-centric algorithms, for both sequential and parallel settings, and by leveraging open-source network analysis libraries such as SNAP (http://snap.stanford.edu/) and NetworkX (https://networkx.github.io/), we build an extensible computational framework for mining and analysing PPI networks.

3.1 Architectural overview of the tool

Our framework for analysing PPI networks is built on a distributed system consisting of a set of well-defined units (and services). The framework incorporates a Linux-based architecture with middleware developed with shell-script and C++-based code and scripts. Our network analysis kernels are mostly developed in C++ with MPI libraries. We also have python-based application code and scripts. For job submission, we use moab qsub scripts. All functional units are coupled loosely so as to support extensibility and modifications. Figure 1 depicts the high-level architecture of the framework. We discuss the key components below.

1 *Control unit:* the control unit employs the central communication and coordination mechanism for our tool. It provides asynchronous, loose coupling of the system components. The unit initiates a workflow – put requests for executing jobs. Every analysis task is transformed into a job consisting of an analysis kernel. Additionally, the control unit facilitates task parallelism by distributing different serial tasks among separate MPI processes. Requests are handled and scheduled by PBS qsub scripts using moab scheduling mechanism. The control unit specifies the details about how a set of analyses is to be fulfilled, in the form of an embedded workflow. An analysis request contains the parameters to run the analysis. The request also contains the specification for the workflow to run, including both pre- and post-processing and inspecting the output. Based on this inspection, a new workflow can be initiated with a new set of parameters and analysis kernels.

2 *Computational resource unit:* once execution requests are identified, they are run on a specific physical machine. It is done by constructing system-specific job submission scripts and monitoring the progress of the execution. To achieve larger scalability, we need to speed up the analysis significantly and make use of the computing clusters efficiently. We design MPI-based parallel computing techniques to scale our methods to large networks and to a large number of processors. Our motif counting methods are based on efficient MPI-based algorithms (Arifuzzaman et al., 2013). To execute a bunch of sequential analysis kernel, we design task parallelism: we distribute multiple kernels among a set of MPI processes. Since our tool is extensible, new methods (either serial or parallel) can easily be integrated. Our scripts automatically assign them to appropriate number of processors guided by the metadata of the executable method.

3 *Analysis unit:* analysis unit is the computational engine behind mining PPI networks. This unit consists of scalable network analysis kernels, both the ones developed from scratch for this tool and from open-source graph analysis algorithms. Since the description of this unit is rather involved, we present it in the next section separately. In conjunction to analysis unit, we have a *data management sub-unit*: this unit is responsible for managing the data resources that reside on a system. The unit also deals with transferring non-local datasets, cleaning datasets, applying scores/thresholds, converting formats, storing or formatting results, etc. There are several high performance services developed for data management. For instances, we implement parallel read, where processors can read disjoint portions of a file in parallel.

4 *Data report or visualisation unit:* our report and visualisation unit is based on gnuplot tool (http://www.gnuplot.info). We generate numerous

statistics plots and distribution using gnuplot. Such capability is integrated with analysis unit, so generation of these tools are automated. Adding a new plot and visualisation capability is straightforward and requires little C++ coding. A new visualisation is modularised (and thus flexible and easy to maintain) by the virtue of being a C++ object.

We also use a java-based visualisation library *Gephi* (https://gephi.org/) for generating additional visualisations. Gephi is open source, modular, and easily extensible through plugins. It is also rich in visualisation features. To create a visualisation of a network, the network is converted into gexf format, an XML representation. The format allows for dynamically adding multiple attributes to nodes and edges. Any layout algorithms can be used to determine object locations. Statistics such as betweenness, page rank, and degree can be applied to decide the size and colour of the nodes and edges. Visualisation by Gephi can give useful insights into a network by highlighting important nodes, edges and communities in a graph or a subgraph. The primary features and benefits of such visualisation are as follows.

- Convenient layouts: Gephi provides several layout algorithms from the literature such as Force Atlas, Yifan Hu and Fruchterman Reingold (https://gephi.org/).
- Feature-based organisation: the node sizes can be proportional to their degrees, betweenness centrality, or other network metric.
- Subgraph visualisation: it offers visualisation of subgraphs that is very useful, especially for massive networks. We have developed several heuristics for choosing subgraphs. First, find a seed (by random seed, central nodes, etc.). Second, expand the seed by a BFS traversal.

Using Gephi orthogonal to gnuplot gives the user additional capabilities for visual analysis. The inputs and parameters needed for Gephi is automatically computed by our tool. The user can interact with the tool to configure different visualisations. Note that our framework allows for adding any open source visualisation tool with little coding effort.

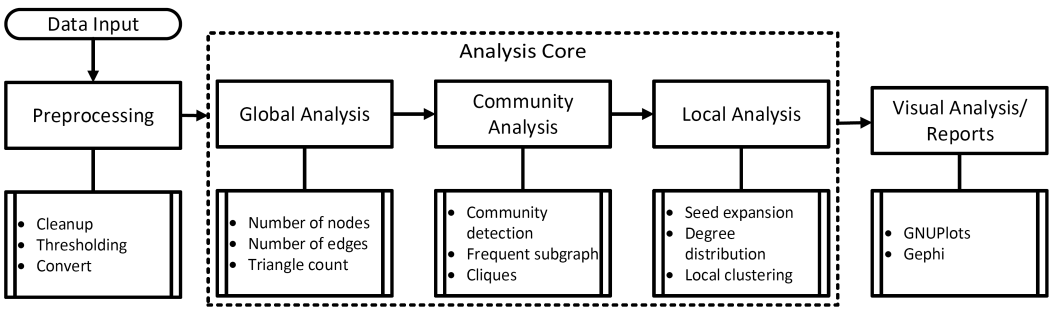
4 Network analysis kernels

A suite of graph metrics (or analysis kernels) is used as the computational engine behind our framework. These kernels are of varying levels of complexity and computational intensity. We classify them into three categories based on the topological granularity they focus on – global, community, and local, as shown in Figure 2. Note that our framework is readily extensible to include any graph kernels. Further, how many of these kernels will be used for a particular investigation depends on the requirements of the analysts.

We use the global metrics to measure the high-level properties of the PPI networks. These metrics are mostly less expensive and are intended to work on the entire graph. For more expensive and complex measures, we use parallel implementation of them. As instance, our tool adapts the parallel algorithms presented in Arifuzzaman et al. (2013, 2015a, 2015b) to find signed triangular motifs at scale. These algorithms are based on efficient partitioning and load balancing schemes and scale to large networks.

We use another suite of metrics to investigate PPI networks at community level. Complex systems are organised in clusters or communities, each having a distinct role or function. In the corresponding network representation, each community appears as a dense set of nodes having higher connection inside the set than outside. Communities reveal the organisation of complex systems and their function. For PPI networks, a community is often interpreted as a functional unit, and thus, community detection is also another important analysis kernel for PPI networks. We use several scalable algorithms for community detection such as Louvain (Blondel et al., 2008) and label propagation (Raghavan et al., 2007). We also use several related analysis kernels such as k-core decompositions. Such decompositions can leverage the higher-order structures to locate the dense subgraphs with hierarchical relations.

Figure 2 Schematic diagram of our analysis workflow starting from data preprocessing to the generation of reports and visualisation



Note: The workflow supports a multi-level approach with a variety of analysis kernels working on different topological granularity

Computation on individual nodes is done by using local metrics. Local metrics are usually the slowest among the kernels. We implemented several distribute-memory algorithms such as computing local clustering coefficients and local Jaccard indices. We are also in the process of adding more parallel kernels. Serial analysis kernels can also be used using task parallel execution as discussed in Section III. Further, it is also an attractive option to first identify important subgraphs by community analysis and then apply the local metrics on the subgraphs (which is smaller than the original graph). Centrality metrics such as local between centrality and closeness centrality are also important local metrics for identifying central nodes of bio-chemical significance.

4.1 A multi-level approach

Our workflow suggests a multilevel approach for efficient analysis: It is generally advised to start analysis with the coarsest (global) and becoming finer at each iteration. Any structure identified as interesting at a coarse level is passed down to be analysed at the next finer level. We generally identify three levels, based on the topological granularity levels, as mentioned above as global, community, and local levels. At the coarsest level, only the global metrics can be applied on the whole network. Communities and local metrics on individual nodes are not considered at this stage. We use efficient and scalable global metrics. Next, community-level metrics are computed. Individual communities can then be locally analysed by applying local metrics. Note that such multi-level approach allows to work with even very scare resources (a commodity laptop) in a computationally efficient way. However, our parallel algorithms and scalable HPC-based framework allows to apply local metrics on the entire networks. Hence the analysts are not limited to follow the multi-level approach in a strict order, rather the approach serves as an organisational or workflow suggestion.

As for the analysis automation, a simple self-descriptory script serves as the starting point of the workflow. It is straightforward to specify the analysis kernels and input network to work on. After initialising the workflow, all the remaining steps such as data pre-processing, analysis, and generation of reports and plots are fully automated. The end-user can inspect the reports and plots and then re-run analyses with different parameters and kernels, if needed.

5 Experimental results and implications

We provide a flexible tool to support scalable data analytics for PPIs. The tool reveals useful patterns and properties from PPI networks by using appropriate mining and analysis techniques. We present a summary of computed network metrics, their biological relevance, scalability of the tool, and a comparison with previous tools below.

5.1 Computing global network metrics

Our global analysis consists of metrics such as finding general statistics (e.g., number of edges, nodes), finding patterns and motifs, e.g., counting triangles, and finding diameter of the networks. Table 1 shows the number of proteins and interactions for five PPI networks. Homo Sapiens dataset has a large number of proteins and their identified interactions. Albugo Laibachii dataset also has over a million protein interactions. We present several analyses on all five datasets of Table 1.

5.1.1 Finding patterns or motifs

Network motifs of size 3 and 4 are overrepresented in real-world networks generated through processes such as hyperlink creation, language formation, and personal social network propagation. Such structures in biological functional networks are suggestive of processes such as positive and negative feedback loops (Ewing et al., 2007; Schwikowski et al., 2000), which have important implications for therapeutic strategies. We enumerate signed triangles for networks datasets of Table 1. As shown in Table 2, Homo Sapiens and Albugo Laibachii networks have 321.6M and 215.12M triangles, respectively, which indicates a high triangle density (triangles per node). In fact, Algugo Laibachii has the highest triangle density among the five datasets. Table 2 also shows average clustering coefficients (CC) of the five datasets. These values are large, indicating the proteins interact with the neighbourhood quite closely.

Table 2 Network properties of our datasets: degree, components, coreness, triangles, clustering coefficients (CC), and diameter statistics

Networks	*Degree*			*Components*		*Max. k-core*	*Triangles*	*Avg. CC*	*Diameter*
	Min.	*Max.*	*Avg.*	*# of comp.*	*Max. size*				
Acetobacterium Woodii	1	2,075	172.51	1	4,192	146	6.26M	0.191	6
Albugo Laibachii	1	2,676	493.44	21	5,798	566	215.12M	0.476	6
Bacillus Cytotoxicus	1	1,746	159.51	2	3,803	146	6.41M	0.226	5
Dinoroseobacter Shibae	1	2,371	229.04	1	3,574	172	13.06M	.297	5
Homo Sapiens	1	10,853	444.12	1	19,247	791	321.6M	0.231	6

5.1.2 Computing diameters

We compute diameters to find insights about reachability and ease of communication and diffusion in PPI networks. The diameters are less than 6 for all networks (Figure 2), suggesting good reachability in the network. Any biochemical process originating in a particular protein can reach to the farthest protein in only six hops. Domain experts may find this information useful in designing drugs for target proteins. Our implementation of diameter kernel is adapted from SNAP library (http://snap.stanford.edu/).

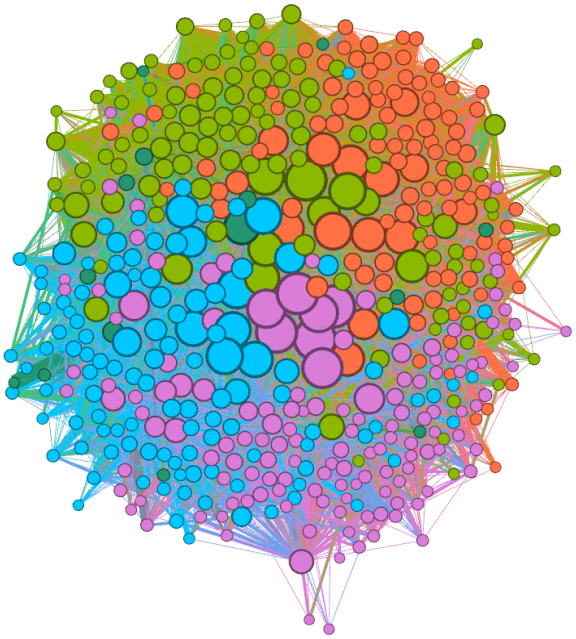
5.2 Community and subgraph-based analysis

We execute community detection methods to reveal functional units in PPI networks. The community statistics are shown in Table 3. For all PPI networks, a number of functional units are detected: for example, for Homo Sapiens, five different functional units (group of proteins) are revealed by our community analysis. The modularity scores quantify the degree of cohesiveness (tightly coupledness) of protein in these communities. We can further inspect the community structures visually with Gephi, as shown in Figure 3. Gephi supports interactive visualisation– for example, the neighbourhood of a particular node can be zoomed in and inspected for details. Further, we also decompose the graph into different connected components, when available, to find their properties. The component statistics reveal whether the network consists of a single or multiple connected component, as shown in Table 2. For example, Homo Sapiens has a single connected component, whereas the Albugo Laibachii network consists of several components. Another important neighbourhood and subgraph based metric is k-coreness. We also investigate kcores of different networks. Table 2 reports the maximum coreness for each of the five PPI networks. Homo Sapiens has a maximum coreness of 791 – it has a subgraph where each node has degree at least 791. This indicates a large cohesive group. Figure 4 shows k-core distribution of several PPI networks. K-core decompositions can leverage the higher-order structures to locate the dense subgraphs with hierarchical relations.

Table 3 Community statistics of five PPI networks of our datasets

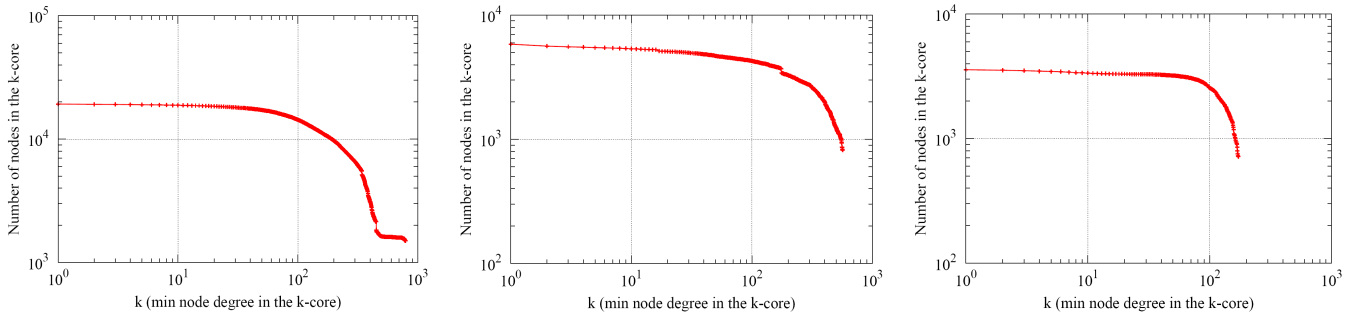
Networks	*Comm. size*		*# of comm.*	*Modularity*
	Max.	*Avg.*		
Acetobacterium Woodii	1,721	419	10	0.170
Albugo Laibachii	2,281	739	9	0.159
Bacillus Cytotoxicus	1,441	317	12	0.135
Dinoroseobacter Shibae	1,173	595	6	0.129
Homo Sapiens	7,296	4,013	5	0.207

Figure 3 Community structure in a subgraph of Homo Sapiens PPI network



Note: Node colours are based on community membership and node sizes on degrees. The plot is generated by Gephi and can further be interactively investigated.

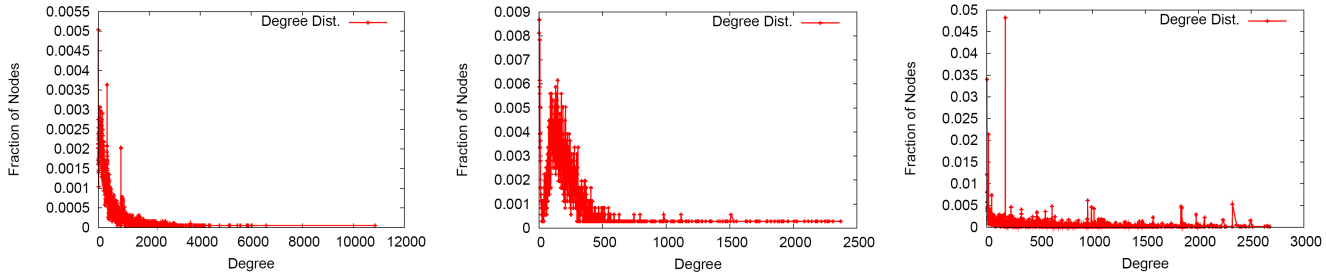
Figure 4 Kcore distribution of three PPI networks, (a) Homo Sapiens (b) Dinoroseobacter Shibae (c) Albugo Laibachii (see online version for colours)



(a) (b) (c)

Note: Coreness is suggestive of the existence of cohesive group and neighbourhood. All the above networks have large coreness consisting of a large portion of nodes.

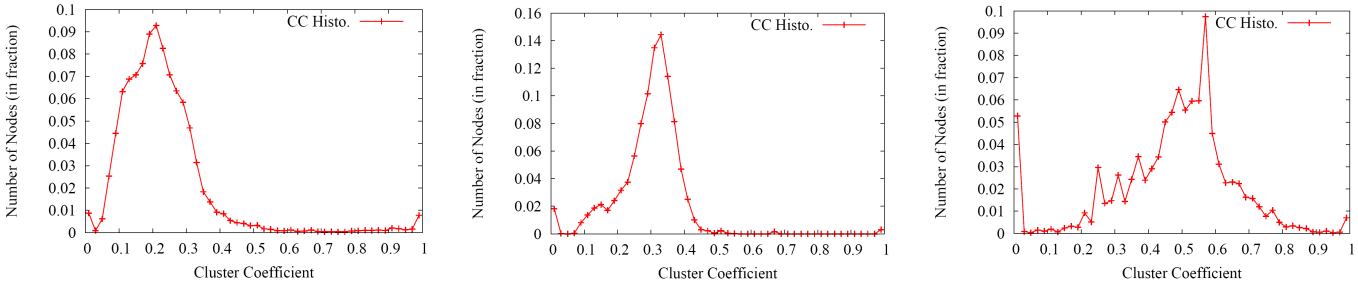
Figure 5 Degree distribution of three PPI networks, (a) Homo Sapiens (b) Dinoroseobacter Shibae (c) Albugo Laibachii (see online version for colours)



(a) (b) (c)

Note: There are few nodes with large degress. However, most of the nodes have small degrees.

Figure 6 CC histogram of three PPI networks, (a) Homo Sapiens (b) Dinoroseobacter Shibae (c) Albugo Laibachii (see online version for colours)



(a) (b) (c)

Note: Most nodes have the clustering coefficients around the global average.

5.3 *Analysis of local metrics*

We computed several local metrics such as clustering coefficient (CC) on nodes, degree distribution, expanding the neighbourhood of a node (seed expansion), to find properties on individual nodes. Figure 5 shows that all networks have few high degree nodes whereas most of the nodes have small degrees. Figure 6 shows the CC distribution of three PPI networks. Most of the nodes (proteins) have clustering coefficients centred around the global average, even though a small percentage of nodes have large clustering coefficients. Running local metrics can reveal further insights about an individual node and its neighbourhood.

5.4 *Detecting central nodes*

The presence of central 'hub' regulators is a prominent feature in biological networks (Schwikowski et al., 2000). Such nodes make especially attractive drug targets, because they are often central to multiple biochemical pathways involved in processes like cell proliferation (Hopkins, 2008). The case is similar to social networks, where nodes with high centrality can be called *central individuals*, and are important to graph propagation processes, such as gossip (Banerjee et al., 2014). Along the same spirit, we compute various centrality metrics for PPI networks to find influential regions. We present below our experiment on Homo Sapiens dataset for *betweenness*, *closeness* and *degree* centrality.

5.4.1 *Cross-checking central nodes for Homo Sapiens*

We found that the following three proteins have the highest centrality scores for Homo Sapiens: ENSP00000344818 (UBC protein), ENSP00000351686 (PRDM10 protein), and ENSP00000328973 (TSPO protein) (shown in Table 4). The existing literature of PPI supports the importance of the above three proteins. Ubiquitin C (UBC) protein, as its name suggests, is a protein available ubiquitously around the eukaryotic tissues. This explains the higher value of betweenness centrality for this protein. UBC protein is encoded by the UBC gene which regulates cellular ubiquitin levels under stress (Wiborg et al., 1985). UBC protein contributes to liver development and hence, lack of UBC genes in unborn foetuses leads to embryonic lethality (Ryu et al., 2007). PRDM10 is a protein that has been linked to the transcriptional regulation (https://string-db.org/cgi/network.pl?taskId=eU6OEL2pwmaP). Some studies on mice have indicated that this may also help in the development of the Central Nervous System (https://www.ncbi.nlm.nih.gov/gene/56980). TSPO protein, encoded by the TSPO gene, is found in the outer mitochondrial membrane. Generally, TSPO has been linked with cholesterol transport with mixed evidence (Lacapere and Papadopoulos, 2003) and has also been associated with immune response (Pawlikowski, 1993) and heart regulation (Qi et al., 2012) depending on the kind of tissue it is working in.

Table 4 Top three proteins based on centrality metrics

Proteins	*Betweenness*	*Closeness*	*Degree*
ENSP00000344818	0.0798	0.6949	0.5639
ENSP00000351686	0.0094	0.6014	0.3425
ENSP00000328973	0.0082	0.5907	0.3129

5.5 *Detailed analysis on density, connectivity and path*

Based on the earlier results on central nodes and global metrics, we investigated further on density and connectedness. The density for an undirected graph is

$$d = \frac{2m}{n(n-1)},$$

where n is the number of nodes and m is the number of edges. The density is 0 for a graph without edges and 1 for a complete graph. The density of multigraphs can be higher than 1.

Table 5 Graph density of the PPI networks

Networks	*Graph density*
Acetobacterium Woodii	0.0625809
Albugo Laibachii	0.0843773
Bacillus Cytotoxicus	0.0421796
Dinoroseobacter Shibae	0.0648774
Homo Sapiens	0.0230760

Clustering coefficient of node v is computed as follows:

$$C_v = \frac{T_v}{\binom{d_v}{2}} = \frac{2T_v}{d_v(d_v-1)},$$

where T_v is the number of triangles containing node v. The verage clustering coefficient for the graph G is,

$$C = \frac{1}{n}\sum_{v \in V} C_v,$$

where n is the number of nodes in G.

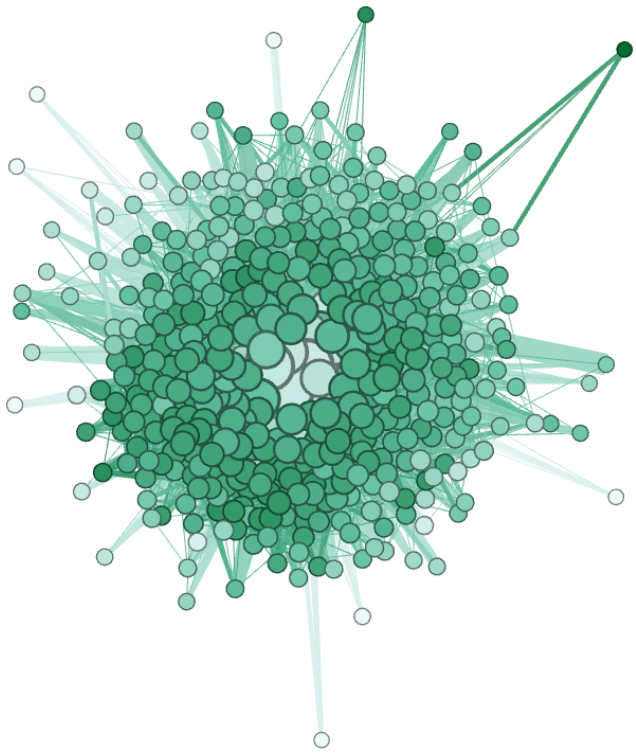
Table 5 demonstrates density of five PPI networks of our datasets. This shows that the Homo sapiens graph has specially lesser density than the other ones and the average clustering coefficient is also lower.

Table 6 Dijkstra's path length between pair of central proteins of Homo Sapiens network

Source	*Destination*	*Path length*
ENSP00000344818	ENSP00000351686	323.0
ENSP00000344818	ENSP00000328973	312.0
ENSP00000351686	ENSP00000328973	300.0

Focusing on the three main proteins from our earlier tests (Table 4), we calculate Dijkstra's path and Dijkstra's path lengths for them between each pair of proteins to find how much they interacted between each other. We also calculate the clustering per node for the three proteins scoring the highest centrality values.

Figure 7 Visualisation of clustering coefficients by node of Homo Sapiens PPI network (see online version for colours)



Note: Node colours are based on the local clustering coefficient (the darker, the higher) and node sizes on degrees. The plot is generated by Gephi and can further be interactively investigated by zooming, stretching, and reshaping.

Table 7 Dijkstra's path between pair of central proteins of Homo Sapiens network

Source	*Destination*	*Path*
ENSP00000344818	ENSP00000351686	344,818, 216,373, 351,686
ENSP00000344818	ENSP00000328973	344,818, 270,570, 328,973
ENSP00000351686	ENSP00000328973	351,686, 323,967, 328,973

Note: Source and destination can be interchanged as we considered undirected paths. The path in column three is given by protein IDs where the prefix ENSP00000 is omitted for brevity.

Table 8 Clustering coefficient per node (local CC) for the central proteins of Homo Sapiens network

Protein ID	*Local CC*
ENSP00000344818	0.0437986
ENSP00000328973	0.0700089
ENSP00000351686	0.0719750

Looking at the clustering per node results in Table 8 (also in Figure 7) and comparing them with an average clustering coefficient in Homo sapiens, we notice that clustering coefficient in these three proteins are comparatively very small. These proteins seem to have fewer clusters around them and calculating and detailing those clusters by domain experts might bear fruitful results.

Dijkstra's path also shows the three other proteins that have low edge weights and associate these proteins with each other. All of the above results might be insightful to a domain expert (e.g., biologist) and we envision that this tool (by generating such results) can be proven useful for multidisciplinary research with large-scale biological data, especially protein interaction data.

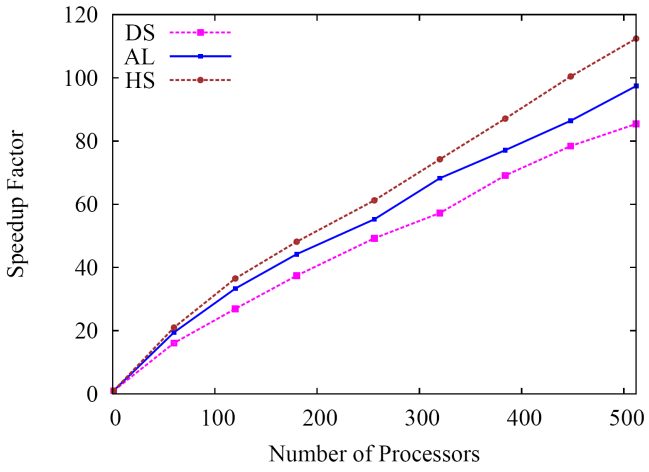
5.6 Scalability analysis

We use scalable algorithmic methods for computing various network metrics. For example, we adapt the methods in Arifuzzaman et al. (2013) to count triangular motifs. The speedup factors for this method on three PPI networks are given in Figure 8. The method shows good speedups and scales almost linearly to a large number of processors. In addition to parallel algorithms, we use efficient sequential methods in a task parallel fashion. We allocate multiple MPI processors and distribute computing kernels among those processors. In effect, this results in a parallel workflow with sequential kernels. Such task parallel design significantly speedup the analysis. As shown in Table 9, our HPC-based workflow achieves almost ten-fold speedup over a serial workflow with ten sequential kernels. Note that this speedup is in excess to what we already achieve with parallel methods such as triangle counting.

Table 9 Workflow scalability: runtime performance for ten analysis metrics with sequential workflow and our HPC-based parallel workflow

Networks	*Runtime (sec.)*		*Speedup*
	Seq. workflow	*Our workflow*	
Acetobacterium Woodii	576	62	9.29
Albugo Laibachii	820	95	8.63
Bacillus Cytotoxicus	540	58	9.31
Dinoroseobacter Shibae	680	72	9.44
Homo Sapiens	1280	130	9.85

Figure 8 Speedup factors of triangle counting algorithm with three PPI networks – Homo Sapiens (HS), Dinoroseobacter Shibae (DS) and Albugo Laibachii (AL) (see online version for colours)



6 Overall capability of the tool, comparisons with others, and future work

The presented tool is new on several levels. The framework builds upon and extends significantly the existing work on scalable algorithms for graph data preprocessing (Arifuzzaman and Khan, 2015) and mining (Arifuzzaman et al., 2013, 2015a), and is intended for big data computation in a scalable and flexible (extensible and sufficiently generic) way. Our tool complements the protein interaction literature with scalable computing methods for efficient analysis and visualisation.

6.1 Big data computation

The volume and variety of the graphs we consider here present real computational challenges in their processing, especially for the weighted and labelled graphs constructed from protein interactions. We utilise distributed systems and parallel computing to develop scalable solutions on the available HPC platforms. We explore parallel algorithms and their implementations as a way to overcome the computational burden placed by the need to process large-scale or high-volume complex networks.

6.2 Future proofing

The above computational framework is extensible we will be able to add new analysis kernels as needed. We also include complex workflow coordination with the framework. Automated reports, plots, and visualisation will be generated from the analyses so that a domain expert can detect interesting patterns, trends, and insights in real time. Based on initial findings, we will be able to adjust the granularity of the analyses and/or network data. We envision that the framework will create a generic computational toolkit for analysing PPI networks. We plan to create a git repository to open the tool for public access.

6.3 Comparison with other network analysis tools

There exist several network analysis tools such as NetworkX (https://networkx.github.io/), Pajek (http://vlado.fmf.uni-lj.si/pub/networks/pajek/), SNAP (http://snap.stanford.edu/), PEGASUS (Kang et al., 2009) and CINET (http://cinet.vbi.vt.edu/granite/granite.html; Abdelhamid et al., 2012). NetworkX is an open source python-based software package for studying complex networks. NetworkX contains a large collection of network algorithms. Pajek is a tool for the analysis and visualisation of networks having thousands to millions of vertices. Stanford Network Analysis Project (SNAP) is a general purpose network analysis library. Another toolkit Network Workbench provides an online portal for network researchers. PEGASUS is a peta-scale distributed graph mining system that provides large-scale algorithms for several graph mining tasks and runs on clouds. CINET is another versatile web-based tool for analysing unlabeled (unsigned) networks.

All the above tools vary in generality, interface, types of networks they support, and the availability of HPC-based resources and frameworks. Many of the above tools, e.g., NetworkX, do not include scalable parallel algorithms or support scalable computing on HPC resources. Some of them, e.g., CINET, lack support for signed networks. Only a few (e.g., CINET) supports workflow coordination. To the best of our knowledge, the novelty of our framework comes collectively from its lightweight, i.e., no need for complex setup or installation of extraneous/expensive support tools, capability to work on signed and weighted networks, offering multi-approach with varying topological granularity, its simple yet efficient workflow coordination, and the availability and incorporation of data and task parallelism through the careful design of distributed-memory algorithms and other HPC techniques. The framework is also extensible and sufficiently generic for many related applications. We are not aware of any previous system that supports HPC-based analytics for PPI networks with such flexibility and efficiency.

We also want to comment that our tool is not a competitor of other existing graph analysis tools. Our tool complements the capabilities of existing tools in several aspects, is extensible, and can integrate many open-source scalable algorithms.

6.4 Future work

If we can avail further data related to drugs, diseases, and protein pathways, together with PPI networks, we will be able to provide a comprehensive case study of the relevance of PPI analytics to drug discovery. Our future work is to demonstrate how drug target discovery can be mapped to network process of a PPI network. For this, as the next step, we will work with drug target data from DrugBank (http://www.drugbank.ca/) database and cancer gene data from the Sanger Institute's COSMIC database (http://cancer.sanger.ac.uk/cosmic). We also plan to initiate collaborations with experts from biological and life sciences so that the results from our tool can further be analysed in light of biomolecular contexts.

7 Conclusions

Interests for PPI networks are growing in biological and medical sciences applications for studying diseases and discovering drugs. The emergence of large volume of PPI datasets challenges efficient and scalable mining of such networks. In this paper, we presented an analytical framework for PPI networks, which addresses the challenges of big data through a flexible tool based on parallel algorithms and other HPC techniques. We demonstrated the scalability and application of the tool on several large PPI networks consisting of millions of edges from a variety of sources. Our tool is effective in identifying central nodes and other interesting patterns. We also introduced different level of analysis granularity to efficiently work with available resources. The tool is also lightweight, flexible, and extensible. We believe that this tool will be useful in tackling emerging large volume of PPI networks (and other related biological networks) and gaining useful insights from them.

Acknowledgements

This work has been partially supported by Louisiana Board of Regents RCS Grant LEQSF (2017-20)-RD-A-25, College of Sciences Internal Grant (University of New Orleans, Spring 2017), and University of New Orleans ORSP Award CON000000002410.

References

Abdelhamid, S.E., Aló, R., Arifuzzaman, S.M. et al. (2012) 'CINET: a cyberinfrastructure for network science', *Proceedings of the 8th IEEE International Conference on e-Science (e-Science 2012)*, October, Chicago, IL, USA, pp.1–8.

Altieri, D.C. (2008) 'Survivin, cancer networks and pathway-directed drug discovery', *Nature Reviews Cancer*, Vol. 8, No. 1, pp.61–70.

Arifuzzaman, S. and Khan, M. (2015) 'Fast parallel conversion of edge list to adjacency list for large-scale graphs', *23rd High Performance Computing Symposium*.

Arifuzzaman, S., Khan, M. and Marathe, M. (2013) 'PATRIC: a parallel algorithm for counting triangles in massive networks', *22nd ACM International Conference on Information and Knowledge Management*.

Arifuzzaman, S., Khan, M. and Marathe, M. (2015a) 'A fast parallel algorithm for counting triangles in graphs using dynamic load balancing', *2015 IEEE BigData Conference*.

Arifuzzaman, S., Khan, M. and Marathe, M. (2015b) 'A space-efficient parallel algorithm for counting exact triangles in massive networks', *17th IEEE International Conference on High Performance Computing and Communications*.

Bader, J.S., Chaudhuri, A., Rothberg, J.M. and Chant, J. (2004) 'Gaining confidence in high-throughput protein interaction networks', *Nature Biotechnology*, Vol. 22, No. 1, pp.78–85.

Banerjee, A., Chandrasekhar, A., Duflo, E. and Jackson, M.O. (2014) 'Gossip: identifying central individuals in a social network', *CoRR*, abs/1406.2293.

Biogrid: Database of Protein, Chemical and Genetic Interactions [online] https: //thebiogrid.org/ (accessed 12 April 2017).

Blondel, V., Guillaume, J., Lambiotte, R. and Lefebvre, E. (2008) 'Fast unfolding of communities in large networks', *Journal of Statistical Mechanics: Theory and Experiment*, No 10, p.10008.

Brastianos, P.K., Carter, S.L., Santagata, S., Cahill, D.P., Taylor-Weiner, A., Jones, R.T., van Allen, E.M., Lawrence, M.S., Horowitz, P.M., Cibulskis, K. et al. (2015) 'Genomic characterization of brain metastases reveals branched evolution and potential therapeutic targets', *European Journal of Cancer*, Vol. 51.

Broder, A., Kumar, R., Maghoul, F., Raghavan, P., Rajagopalan, S., Stata, R., Tomkins, A. and Wiener, J. (2000) 'Graph structure in the web', *Computer Networks*, Vol. 33, Nos. 1–6, pp.309–320.

Chen, J. and Lonardi, S. (2009) *Biological Data Mining*, Chapman & Hall/CRC.

Chiba, N. and Nishizeki, T. (1985) 'Arboricity and subgraph listing algorithms', *SIAM Journal on Computing*, Vol. 14, No. 1, pp.210–223.

Chin, K., de Solorzano, C.O., Knowles, D., Jones, A., Chou, W., Rodriguez, E.G., Kuo, W-L., Ljung, B-M., Chew, K., Myambo, K. et al. (2004) 'In situ analyses of genome instability in breast cancer', *Nature Genetics*, Vol. 36, No. 9, pp.984–988.

CINET System [online] http://cinet.vbi.vt.edu/granite/granite.html (accessed 11 April 2017).

Ensembl Genome Browser [online] http://www.ensembl.org (accessed 2 February 2017).

Ewing, R.M., Chu, P., Elisma, F., Li, H., Taylor, P., Climie, S., McBroom-Cerajewski, L., Robinson, M.D., O'Connor, L., Li, M. et al. (2007) 'Large-scale mapping of human protein-protein interactions by mass spectrometry', *Molecular Systems Biology*, Vol. 3, No. 1, p.89.

Fortunato, S. and Lancichinetti, A. (2009) 'Community detection algorithms: a comparative analysis', *4th International ICST Conference on Performance Evaluation Methodologies and Tools*.

Gephi – The Open Graph Viz Platform [online] https://gephi.org/ (accessed 12 March 2017).

Girvan, M. and Newman, M. (2002) 'Community structure in social and biological networks', *Proceedings of the National Academy of Sciences*, Vol. 99, No. 12, pp.7821–7826.

Han, J-D.J., Bertin, N., Hao, T., Goldberg, D.S., Berriz, G.F., Zhang, L.V., Dupuy, D., Walhout, A.J., Cusick, M.E., Roth, F.P. et al. (2004) 'Evidence for dynamically organized modularity in the yeast protein– protein interaction network', *Nature*, Vol. 430, No. 6995, pp.88–93.

Hopkins, A.L. (2008) 'Network pharmacology: the next paradigm in drug discovery', *Nature Chemical Biology*, Vol. 4, No. 11, pp.682–690.

Kang, U., Tsourakakis, C.E. and Faloutsos, C. (2009) 'Pegasus: a peta-scale graph mining system implementation and observations', *Proc. of the 9th IEEE International Conference on Data Mining*.

Kwak, H. et al. (2010) 'What is twitter, a social network or a news media?', *WWW*.

Lacapere, J-J. and Papadopoulos, V. (2003) 'Peripheral-type benzodiazepine receptor: structure and function of a cholesterol-binding protein in steroid and bile acid biosynthesis', *Steroids*, Vol. 68, No. 7, pp.569–585.

Louisiana Optical Network Infrastructure [online] https://loni.org/ (accessed 2 February 2017).

National Center for Biotechnology Information [online] https://www.ncbi.nlm.nih.gov/genome/viruses/retroviruses/hiv-1/interactions/browse/ (accessed 21 June 2017).

NCBI PRDM10 [online] https://www.ncbi.nlm.nih.gov/gene/56980 (accessed 23 June 2017).

NetworkX Tool [online] https://networkx.github.io/ (accessed 23 March 2017).

Newman, M. (2003) 'The structure and function of complex networks', *SIAM Review*, Vol. 45, pp.167–256.

Pajek Network Analysis Tool [online] http://vlado.fmf.uni-lj.si/pub/networks/pajek/ (accessed 23 March 2017).

Pawlikowski, M. (1993) 'Immunomodulating effects of peripherally acting benzodiazepines', in *Peripheral Benzodiazepine Receptors*, pp.125–135, Academic Press, London.

Qi, X., Xu, J., Wang, F. and Xiao, J. (2012) 'Translocator protein (18 kDa): a promising therapeutic target and diagnostic tool for cardiovascular diseases', *Oxidative Medicine and Cellular Longevity*, Vol. 162934, DOI: 10.1155/2012/162934.

Raghavan, U., Albert, R. and Kumara, S. (2007) 'Near linear time algorithm to detect community structures in large-scale networks', *CoRR*, abs/0709.2938.

Rual, J-F., Venkatesan, K., Hao, T., Hirozane-Kishikawa, T., Dricot, A., Li, N., Berriz, G.F., Gibbons, F.D., Dreze, M., Ayivi-Guedehoussou, N. et al. (2005) 'Towards a proteome-scale map of the human protein–protein interaction network', *Nature*, Vol. 437, No. 7062, pp.1173–1178.

Ryu, K. et al. (2007) 'The mouse polyubiquitin gene UbC is essential for fetal liver development, cell-cycle progression and stress tolerance', *The EMBO Journal*, Vol. 26, No. 11, pp.2693–2706.

Schwikowski, B., Uetz, P. and Fields, S. (2000) 'A network of protein–protein interactions in yeast', *Nature Biotechnology*, Vol. 18, No. 12, pp.1257–1261.

Snap [online] http://snap.stanford.edu/ (accessed 23 February 2017).

Stelzl, U., Worm, U., Lalowski, M., Haenig, C., Brembeck, F.H., Goehler, H., Stroedicke, M., Zenkner, M., Schoenherr, A. and Koeppen, S. et al. (2005) 'A human protein-protein interaction network: a resource for annotating the proteome', *Cell*, Vol. 122, No. 6, pp.957–968.

STRING PRDM10 [online] https://string-db.org/cgi/network.pl?taskId=eU6OEL2pwmaP (accessed 11 April 2017).

String: Functional Protein Association Networks [online] https://string-db.org/ (accessed 17 February 2017).

Suri, S. and Vassilvitskii, S. (2011) 'Counting triangles and the curse of the last reducer', *20th International Conference on World Wide Web*.

Wiborg, O., Pedersen, M., Wind, A., Berglund, L., Marcker, K. and Vuust, J. (1985) 'The human ubiquitin multigene family: some genes contain multiple directly repeated ubiquitin coding sequences', *The EMBO Journal*, Vol. 4, No. 3, p.755.